ENHANCING LEARNER SUPPORT AND MOTIVATION WITH MIND MAP GENERATION

Sri Samadhi L.A.S.S

(IT21302862)

B.Sc. (Hons) Degree in Information Technology Specialized in

Software Engineering

Department of Computer Science and Software Engineering

Sri Lanka Institute of Information Technology

Sri Lanka

April 2025

DECLARATION

I declare that this is my own work, and this proposal does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to Sri Lanka Institute of Information Technology, the nonexclusive right to reproduce and distribute my dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Name	Student ID	Signature
Sri Samadhi L.A.S.S	IT21302862	

The supervisor/s should certify the proposal report with the following declaration.

The above candidates are out research for the undergraduate Dissertation under my supervision.

Signature of the Supervisor Date

04/12/2025

……..... ……..

Dr. Kalpani Manathunga

ABSTRACT

As educational methods continue to evolve, there is a growing emphasis on visual learning tools that align with how our brains process and retain information. Mind maps have emerged as a powerful tool in this context, enabling students to visualize complex concepts and their interrelationships, which enhances comprehension and memory retention. This research focuses on integrating personalized mind maps within a Learning Management System (LMS) to leverage these visual learning benefits effectively.

The proposed system tailors mind map generation based on initial assessments, customizing the maps to reflect individual learning progress and needs. This personalization allows students to interact with the mind maps by expanding or collapsing nodes, which helps them focus on specific areas of their learning journey. The mind maps are further enhanced with links to relevant articles, videos, and external resources, providing a comprehensive learning experience.

In addition to mind maps, the research introduces a digital notecard system for managing key concepts and a note-taking tool integrated with lecture content. A forum feature will also be implemented to facilitate communication between student groups and lecturers, organized by subject for targeted discussions. These interactive tools are designed to improve cognitive processing, support collaborative learning, and ensure students have the resources necessary to succeed in their educational endeavors.

Keywords: *Adaptive Learning, Learning Management System (LMS), Large Language Models (LLM), Return on Investment (ROI), personalized learning, visualization*

ACKNOWLEDGEMENT

First and foremost, I would like to sincerely thank Dr. Kalpani Manathunga, my supervisor, for her unwavering support and advice, which enabled me to complete my research study. In addition to my supervisor, I would like to express my gratitude to Ms. Thilini Jayalath, who is the co-supervisor of this research project, for her willingness to assist whenever assistance was required. Finally, but just as importantly, I would want to thank all of my teammates, family, and friends who have helped this effort either directly or indirectly.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

Abbreviation	Description
AI	Artificial Intelligence
ML	Machine Learning
LMS	Learning Management System
GNN	Graph Neural Network
LLMs	Large Language Models
API	Application Programming Interface
RAG	Retrieval Argument Generation
OER	Open Educational Resources
UI	User Interface
UX	User Experience
ROI	Return on Investment
UAT	User Acceptance Testing
CI	Continuous Integration
CD	Continuous Development

1. INTRODUCTION

1.1. General Introduction

The evolution of technology in education has drastically reshaped the way learning experiences are delivered, offering innovative solutions that enhance student engagement, understanding, and performance. Among the most significant advancements are LMS, which have become central to modern education. These systems provide structured environments where students can access content, track progress, and interact with resources. However, despite the widespread adoption of LMS platforms, many still rely on static, one-size-fits-all approaches that fail to adapt to the unique needs, preferences, and progress of individual learners [1].

A key to enhancing these systems lies in personalizing learning experiences ensuring that content dynamically adjusts to each student's learning style, preferences, and abilities. Mind maps, known for their ability to visually organize and show the relationships between concepts, have proven to be effective tools for improving learning outcomes. Although the concept of mind mapping has existed for centuries, it was in the 1960s that Tony Buzan popularized the technique as a cognitive tool for organizing thoughts [2]. A key historical milestone in mind mapping's application came in 1993, when the technique's use in educational and cognitive sciences saw a significant shift with advancements in neuroimaging. This shift revolutionized how we understand cognitive processing, providing more detailed and accurate analyses of how the brain visualizes and connects concepts. Similarly, mind maps have evolved from simple diagrams to powerful digital tools, becoming integral to modern learning environments. Yet, traditional mind mapping tools within LMS platforms remain static and do not evolve based on a learner’s progress, limiting their potential.

This gap highlights the need for more intelligent systems that can respond to learners' needs in real-time. By integrating AI technologies, such as RAG and LLMs, it is possible to create adaptive, learner-centric mind maps that evolve in tandem with a student’s knowledge and performance. This approach promises to personalize the

learning journey, offering real-time feedback and dynamically updated mind maps, ultimately improving student engagement and retention.

Despite significant advancements in educational technologies, disparities persist in the accessibility of personalized learning systems, especially in underdeveloped regions and underserved populations. In many areas, uneven access to resources, instructors, and technology continues to limit the potential of education. In countries with limited access to advanced educational infrastructure, the absence of adaptive learning solutions exacerbates educational inequalities. The growing global demand for scalable educational systems, driven by an increasing student population and evolving learning needs, only amplifies this challenge. As such, the need for adaptive learning platforms that cater to diverse learning styles and paces has never been more urgent.

With the rise of AI-driven technologies, the development of automated and adaptive LMS has garnered significant attention. These systems can dynamically adjust content and learning paths based on a learner's individual progress, ensuring a personalized educational experience. By employing advanced machine learning models like RAG, LMS can make real-time adjustments, including personalized mind map generation and other learning tools. In addition, personalized resources and linked references can be incorporated into each mind map node to provide further clarity and depth for students. However, integrating these technologies into existing systems, particularly in regions with limited access to quality education, presents significant challenges. It is crucial to develop robust, accessible, and efficient LMS platforms that can be implemented across various contexts to enhance educational quality.

The integration of AI and ML into LMS platforms provides a promising solution to these challenges. Leveraging AI enables the creation of personalized learning paths that adapt in real-time to student progress. A key innovation of this research is the integration of AI-driven mind map generation, which allows students to visualize and organize learning materials according to their unique understanding. This personalized approach supports students in identifying relationships between concepts, enhancing comprehension, and improving retention. Additionally, by linking relevant academic

resources and generating visual representations for each node, the system fosters a deeper understanding of topics.

Building such a system, however, presents challenges. AI models must accurately interpret vast amounts of student data to generate effective learning materials. Additionally, integrating AI and mind mapping technologies into existing LMS infrastructure requires thoughtful design, extensive validation, and continuous optimization to ensure the system's effectiveness and user-friendliness.

Despite these challenges, the potential benefits are vast. This research aims to develop an innovative AI-driven Adaptive Learner-Centric LMS that incorporates personalized mind map generation to enhance student engagement with resource linking, learning efficiency, and overall academic performance. By integrating these technologies, the system will provide each student with a tailored, dynamic learning environment that fosters deeper understanding and improved learning outcomes.

1.2. Literature Review

The integration of AI in educational technology has significantly transformed the design of personalized learning systems. A central element of this transformation is the development of adaptive mind maps, which visually organize knowledge and enhance understanding and retention. These AI-powered mind maps dynamically adjust based on a learner's progress, creating personalized learning experiences that promote greater engagement and efficiency. This section reviews key research studies on AI-driven adaptive mind map systems, focusing on techniques such as LLMs, RAG, and knowledge graph generation.

Razafinirina et al. (2024) extensively explore the role of LLMs in personalized learning, particularly their pedagogical alignment within educational settings [3]. Their study highlights how LLMs can generate content that adapts to the unique needs of individual learners in real-time, facilitating error correction and question answering. While LLMs offer tailored responses, the authors caution that excessive personalization may lead to cognitive overload, emphasizing the need for a balance between customization and cognitive load. This concern is vital in the development of adaptive learning systems, as they must cater to various learning styles without overwhelming the learner. Moreover, Razafinirina et al. (2024) stress the importance of Knowledge Editing Techniques (KME) in ensuring content accuracy and relevance, preventing errors, and keeping the system's outputs up to date.

In a similar vein, Abbas et al. (2025) investigate the impact of AI-driven concept mapping on reading comprehension in both individual and collaborative learning contexts [4]. Their study reveals that AI-enhanced concept mapping tools improve cognitive engagement and knowledge retention by offering real-time feedback and personalized recommendations. One of the primary advantages of AI-driven concept mapping is its ability to structure complex information and form meaningful connections between concepts. However, Abbas et al. (2025) underscore the importance of adapting these systems to accommodate different learning preferences, as a one-size-fits-all approach may not be effective for all learners. This research aligns with the development of systems that integrate external knowledge sources, such as

search engine APIs, to improve the accuracy and relevance of mind maps, thus ensuring adaptability to the learner's evolving needs.

Traditional mind mapping techniques have often been static, requiring users to manually create diagrams that represent fixed knowledge structures. Recent advancements in AI have sought to automate this process, introducing the flexibility to adapt mind maps in real-time based on learner interactions. For instance, Hu et al. (2021) developed a sequence-to-graph model that utilizes GNNs to identify relationships between concepts within textual documents [5]. This model enhances the visual representation of knowledge structures by accurately mapping out the connections between ideas. Despite its strengths, Hu et al. (2021) acknowledge that their approach does not consider changes in the learner's understanding over time, as the mind map remains static once generated [5]. This limitation has been addressed in subsequent studies, which have introduced dynamic adaptability into mind maps based on real-time learner assessment data, marking a significant advancement in personalized learning.

Kudelić et al. (2012) contributed to this field with a system that employs text-mining algorithms to extract key concepts from educational texts and generate structured mind maps [6]. While this approach represented a notable step forward in automating mind map creation, it was hindered by its static nature and inability to adapt to individual learner progress. Later systems overcame this limitation by integrating RAG and LLMs, allowing mind maps to evolve based on learner feedback, such as quiz results, which reflect the learner's strengths and weaknesses. This dynamic approach to mind map generation signifies a shift toward more personalized learning environments that adapt to the learner's ongoing development.

In addition to these advancements, knowledge graphs have emerged as a powerful tool in AI-driven mind mapping. Bae et al. (2020) explored the potential of knowledge graphs to uncover unexpected relationships between concepts, fostering creativity and problem-solving [7]. However, they note the computational complexity involved in implementing knowledge graphs and their limited capacity for personalization in dynamic learning contexts. To address these challenges, some systems have

incorporated external resources, such as APIs from search engines, to provide real-world context and enrich generated mind maps with additional materials like videos, articles, and external references. These resources enhance the mind maps by situating the concepts within a broader learning context, making the learning experience more comprehensive and engaging.

The studies reviewed illustrate the growing potential of AI-based adaptive mind mapping systems in personalized learning. However, challenges remain, particularly regarding the accurate generation of personalized content, the need for real-time adaptability, and the management of large-scale data. The integration of LLMs, RAG, and external knowledge sources presents a promising solution to these challenges, enabling the creation of adaptive mind maps that evolve based on learner performance and engagement. These advancements pave the way for scalable, effective, and contextually relevant learning experiences in diverse educational settings.

Future research is likely to focus on refining content generation efficiency, minimizing cognitive overload, and enhancing real-time adaptability. As AI technology continues to progress, these systems will become increasingly adept at addressing individual learning needs, making education more engaging, personalized, and aligned with each learner’s unique progression.

1.3. Research Gap

There have been several studies related to the automatic generation of mind maps, their personalization, and real-time adaptability. However, there remain critical gaps in the research, specifically in areas such as dynamic generation, personalization, integration of multimedia sources, and real-world applicability. The following table summarizes the comparison of relevant studies and their contributions to the identified gaps.

Table 1: Research Comparison

Research	Dynamic Mind Map Generation	Personalization of Mind Maps	Multimedia Integration	Real-World Application	Scalability
Research A [8]	☑	✗	✗	✗	✗
Research B [9]	✗	☑	✗	✗	✗
Research C [10]	✗	✗	☑	✗	✗
Proposed System	☑	☑	☑	☑	☑

Research A, conducted by Elhoseiny and Elgammal (2016), explores a novel method for hierarchical visual abstraction of natural language text, transforming text into multi-level mind maps [8]. The study presents an approach that visualizes text documents across multiple levels of abstraction, with higher-level mind map nodes representing abstract information and child nodes capturing more detailed concepts. The method uses Detailed Meaning Representation (DMR) to facilitate the transformation of text into mind map visualizations. However, this approach focuses primarily on static mind map generation, which lacks real-time dynamic adaptability

based on evolving content or learner performance. Additionally, the method does not account for individual learner preferences or incorporate multimedia elements such as images or videos. Although this research offers a foundational approach to automated mind map generation, it does not fully address the need for personalized, adaptive, and multimedia-enriched mind maps in dynamic learning environments.

Compared to the proposed system, this research focuses on static mind map generation, lacking real-time adaptability, personalization, and multimedia integration. In contrast, the proposed system utilizes RAG and LLMs to dynamically generate personalized mind maps based on learners' quiz results and preferences. Furthermore, the proposed system integrates multimedia content, enriching the mind maps and providing a more engaging and interactive learning experience.

Research B, conducted by Shi et al. (2023), investigates the effectiveness of mind mapping-based instruction on students' cognitive learning outcomes through a meta-analysis of 21 studies [9]. The study concludes that mind mapping-based instruction has a more positive influence on students' cognitive learning outcomes compared to traditional methods. It also identifies that lower-grade students are more receptive to mind mapping-based instruction, particularly in subjects like Science, Technology, Engineering, and Math (STEM). However, while the study highlights the advantages of mind mapping in enhancing learning outcomes, it does not address the limitations of traditional mind mapping methods, such as the lack of dynamic adaptation to individual learning needs. Moreover, the study focuses primarily on instructional effectiveness and does not explore how mind maps can be personalized based on real-time learner data, such as quiz results or cognitive progress.

Compared to the proposed system, Research B emphasizes the effectiveness of mind mapping in educational contexts but fails to incorporate real-time adaptive elements that respond to individual learner progress or multimedia integration. In contrast, the proposed system utilizes advanced techniques like LLMs and RAG to dynamically adjust mind maps based on the learner's performance and needs. Additionally, it integrates multimedia elements such as images and videos, which enrich the learning experience by making the mind maps more interactive and engaging.

Research C, authored by Okada and Connolly (2008), explores the use of knowledge maps in the development of OERs to enhance meaningful learning [10]. The study focuses on the Open Learn project, which uses the Compendium software tool for visual thinking to connect ideas, concepts, and resources. The research emphasizes how knowledge mapping can help condense large volumes of information, visualize connections between learning materials, and facilitate new insights. However, while the study highlights the benefits of visualizing knowledge using mind maps, it primarily addresses static representations of learning materials and does not explore dynamic or personalized mind map generation. Furthermore, the study does not incorporate multimedia sources as input for mind map creation or address the challenges of implementing such systems at scale in large educational environments.

Compared to the proposed system, Research C contributes to the integration of knowledge mapping for OERs but lacks dynamic adaptation based on learner data and the ability to process multimodal input, such as images and videos. The proposed system, on the other hand, incorporates multimodal data processing, allowing for the generation of adaptive mind maps based on both textual and multimedia sources. Additionally, the system is designed for real-world implementation, offering scalability through cloud-based solutions, ensuring efficient performance across large learner populations.

1.4. Research Problem

The current landscape of mind map generation presents several critical limitations that hinder its potential to enhance learning experiences effectively. Traditional methods predominantly rely on manual input, which is not only time-consuming but also susceptible to human error and inconsistency. These manual approaches lack the automation necessary to accommodate the complexity of diverse learning materials, often resulting in poorly structured or incomplete mind maps that diminish their educational value.

A central challenge lies in the absence of intelligent, automated systems capable of dynamically generating mind maps tailored to a learner's evolving needs. Without

real-time integration of user data such as quiz performance or cognitive engagement existing systems cannot adapt content complexity or structure in response to individual progress. This significantly restricts opportunities for personalized learning experiences.

Furthermore, current mind mapping tools tend to support only text-based content, overlooking the integration of multimedia inputs like images, videos, and audio. In today's rich digital learning environments, such media are essential for supporting various learning styles and improving learner engagement. The lack of multimedia support not only narrows the applicability of mind maps but also fails to address diverse cognitive and perceptual preferences among learners.

Another substantial limitation is the absence of frameworks that incorporate individual learner differences, such as prior knowledge, learning preferences, and cognitive load. Without considering these factors, existing systems fall short in delivering mind maps that are adaptive, personalized, and educationally meaningful. Additionally, while connecting mind maps to external resources can enrich the learning experience, the quality, credibility, and relevance of these linked contents are often unregulated or unverified posing a risk to information integrity.

Scalability also remains a pressing concern. Implementing adaptive mind map systems for large learner populations demands significant computational resources, which are often not optimized in current solutions. Moreover, the lack of longitudinal studies investigating the sustained impact of adaptive mind maps on learner retention and engagement leaves a critical gap in understanding their long-term educational value.

Given these challenges, there is a clear and pressing need for research focused on developing intelligent, automated mind map generation systems that leverage advanced technologies. Our system is capable of generating personalized, multimedia-enriched, and real-time adaptive mind maps that align with learners' evolving needs. Addressing these gaps will be instrumental in advancing the practical deployment of adaptive mind mapping tools across diverse and scalable educational environments.

2. RESEARCH OBJECTIVES

2.1. Main Objectives

The primary objective of this research is to develop an intelligent system for automated mind map generation that adapts to various learning content and individual learning needs. The system will automatically generate effective mind maps from diverse input types, including text, images, and multimedia sources. These mind maps will be personalized based on users' specific learning styles, thereby enhancing knowledge retention, understanding, and the overall learning experience.

2.2. Sub Objectives

• Automated Extraction of Key Concepts

The primary objective is to develop a robust mechanism for automatically extracting key concepts and their relationships from diverse content sources. This process will identify critical information, such as main ideas, sub-concepts, and interconnections, ensuring that the system accurately represents the content’s structure in a mind map format without requiring manual intervention.

• Personalization Based on Learning Styles

The system will adapt to individual learning styles by analyzing user preferences and customizing mind maps accordingly. By leveraging LLMs, such as GPT models, the system will identify weak areas based on users’ quiz performance and generate personalized mind maps that align with their cognitive profiles. This approach enhances engagement, comprehension, and knowledge retention by addressing specific learning gaps and adapting content dynamically.

• Multimedia Resource Integration in Mind Map Nodes

A key objective is to integrate multimedia elements, such as images and videos, within mind map nodes. The system will automatically incorporate relevant multimedia content to enrich the mind maps, providing a more dynamic and interactive

representation of information. This feature aims to support diverse learning styles including visual, auditory, and kinesthetic by accommodating various content formats and enhancing the overall learning experience.

• Mind Map Generation via Prompt Engineering

The system will enable users to generate mind maps based on custom prompts, allowing for flexibility in content creation. Users can input any topic or requirement, and the system will generate a structured and informative mind map accordingly.

• Personalized Mind Map Storage and Management

To enhance personalized learning, the system will allow users to maintain up to three updated mind maps. If a user saves a new mind map after completing a quiz while already having three stored, the earliest saved mind map will be automatically replaced. This ensures that users always have access to their most recent and relevant learning materials.

• Development of a User-Friendly Interface

The system will feature an intuitive and user-friendly interface tailored for students and educators. The interface will facilitate easy content input and customization of mind maps to align with individual learning goals. Additionally, visualization tools, such as zooming and reorganization options, will be incorporated to improve usability and comprehension.

• Real-Time Mind Map Generation and Optimization

To enhance efficiency, the system will be optimized for real-time mind map generation, enabling users to receive instant feedback and visual representations of their content. This feature ensures a seamless, interactive, and adaptive learning experience.

• Deployment and Scalability Using AWS and Docker

The system will be deployed on Amazon Web Services (AWS) to ensure high availability, scalability, and performance. Docker will be utilized for containerization, enabling efficient deployment, portability, and seamless scaling of the system across different environments.

• Evaluation of System Performance

The system's accuracy, efficiency, and effectiveness will be rigorously evaluated. Testing will be conducted on diverse datasets and real-world scenarios to assess its ability to process different content types and adapt to various learning contexts. Key performance indicators (KPIs) such as user satisfaction, learning outcomes, and time efficiency will be used to measure overall system performance.

• Integration into Educational Systems

A long-term goal is to integrate the mind map generation tool into existing educational platforms, ensuring seamless adoption by students and educators. Additionally, the system will be designed to support collaborative features, allowing multiple users to work on shared mind maps in real-time, promoting teamwork and knowledge sharing.

3. METHODOLOGY

Our Adaptive Mind Map Generation System for LMS offers significant value to learners, educators, and academic institutions by promoting personalized and efficient study methods. Students who wish to better understand their learning materials or identify areas needing improvement can interact with the platform through an easy-to-use web interface. By uploading learning contents, answering quizzes, or providing custom prompts, the system can generate personalized mind maps to match each learner's current needs and progress.

The mind maps are created using intelligent algorithms that combine LLMs, RAG, and multimedia data processing techniques. These technologies allow the system to understand and extract key concepts, generate visual connections between ideas, and tailor content based on individual performance. The system also supports prompt-based mind map generation, where users can input custom queries or topics and receive structured mind maps linked to the original content.

Although the system provides real-time and personalized insights for self-paced learning, it is important to note that it is a supportive educational tool and does not replace a teacher’s guidance. By bridging traditional study methods with AI-driven tools, this solution empowers students to take a more active role in their learning journey, leading to improved understanding, engagement, and academic outcomes.

3.1. Requirement Gathering and Analysis

- Survey-Based Requirement Gathering from University Students Across Sri Lanka

To understand the key challenges faced by students and their preferences regarding learning methodologies, a comprehensive survey was conducted across various universities in Sri Lanka. The survey aimed to gather insights into students' study habits, their use of visual learning tools, and the effectiveness of external learning resources.

The survey responses highlighted the following critical aspects:

- A strong preference for visual learning techniques, particularly mind maps, as an effective tool for knowledge retention and comprehension.
- A demand for integrating external resources, such as academic articles, videos, and supplementary learning materials, to enhance the learning process.
- The need for personalized learning pathways that adapt based on individual strengths and weaknesses, particularly through automated feedback mechanisms.
- A requirement for a user-friendly and interactive system that enables seamless mind map generation, storage, and retrieval**.**

3.2. Requirement Analysis

Based on the gathered insights, a thorough requirement analysis was conducted to define the key functionalities of the AI-driven Adaptive Learner-Centric LMS. The system is designed to address the identified gaps in current learning methods by implementing:

- Automated Extraction of Key Concepts – The system will autonomously extract essential topics, concepts, and relationships from learning materials to generate structured mind maps.
- Personalized Learning with AI – By leveraging LLMs, the system will analyze students' quiz performance, identify weak areas, and generate customized mind maps to enhance understanding.
- Integration of External Learning Resources – The platform will dynamically incorporate Google Search API, PDFs, videos, and other academic materials into mind map nodes, enriching the learning experience.
- Interactive and User-Friendly Interface – A well-designed interface will allow students to interact with mind maps easily, ensuring accessibility and usability.

3.3. Project Requirements

After analyzing the gathered requirements and defining the main objective along with the specific sub-objectives, the following requirements were identified for implementation in the proposed AI-driven Adaptive Learner-Centric LMS.

3.3.1. Functional Requirements

- Automated Concept Extraction: The system should automatically extract key concepts and their relationships from various input sources, such as PDFs, online resources, and lecture notes.
- Personalized Mind Map Generation: The platform should analyze quiz performance and learning behavior using LLMs to generate personalized mind maps that cater to individual learning needs.
- Dynamic Content Integration: The system must incorporate external learning resources (e.g., Google Search API, PDFs, multimedia) into mind map nodes to enrich the learning experience.
- Real-Time Mind Map Customization: Users should be able to generate mind maps based on custom prompts through prompt engineering, allowing flexible content structuring.
- Mind Map Versioning & Management: The system should allow users to store up to three personalized mind maps. If a new mind map is generated when three are already stored, the oldest mind map should be replaced automatically.
- Interactive Visualization: Users should be able to manipulate the generated mind maps, expand or collapse nodes, and zoom in/out for better comprehension.

3.3.2. Non-Functional Requirements

Performance

- The system must generate mind maps and retrieve relevant learning resources in real-time to ensure a smooth user experience.
- The system should efficiently handle a large number of concurrent users without performance degradation.

- The system architecture should be scalable to support increasing data volumes and user demands while maintaining optimal performance.

Reliability

- The system must ensure high availability (99% uptime) to support uninterrupted access for students and educators.
- The platform should be designed to gracefully handle failures, ensuring critical functions remain operational even in case of system errors.
- Automated backup mechanisms should be in place to prevent data loss and ensure seamless recovery.

Security

- User data, including quiz results and personalized mind maps, must be encrypted while stored and transmitted to prevent unauthorized access.
- The system must implement strong authentication mechanisms (e.g., multi-factor authentication) and access control policies to restrict sensitive data access.
- All communication between clients (web application) and the server must be encrypted using industry-standard protocols (SSL/TLS) to ensure secure data transmission.

Usability

- The UI should be intuitive, visually appealing, and easy to navigate, enabling students and educators to interact with the system effortlessly.
- The system should provide clear instructions, tooltips, and feedback messages to guide users through different functionalities.
- Mind maps should be interactive, customizable, and accessible to support a variety of learning preferences.

Maintainability

- The system architecture should follow a modular and microservices-based approach, facilitating easy maintenance, updates, and scalability.
- Comprehensive documentation should be provided, including system operations, API usage, deployment procedures, and error-handling guides for developers and administrators.
- The codebase must adhere to industry-standard coding practices to enable efficient debugging, collaboration, and future enhancements.

Interoperability

- The system should be compatible across multiple devices, browsers, and operating systems to ensure broad accessibility.
- Well-defined APIs should be available to enable seamless integration with other LMS, external content providers, and educational tools.

3.3.3. User Requirements

- Mind Map Generation: Users should be able to generate mind maps by entering prompts, uploading study materials, or integrating quiz results to create personalized learning structures.
- Personalized Mind Maps: The system should allow users to store and update up to three personalized mind maps. When a new mind map is generated after reaching the limit, the earliest stored mind map should be replaced.
- Quiz-Driven Adaptation: The system should analyze quiz results and automatically update mind maps based on user performance, highlighting weak areas and reinforcing important concepts.
- Multimedia Integration: Users should be able to view relevant images, videos, and external links within mind map nodes to enhance the learning experience.
- Seamless Accessibility: The system should be accessible on web application, ensuring a consistent and responsive user experience across different devices and screen sizes.

- User Authentication & Authorization: Users should be able to register, log in, and securely access their personalized learning content, ensuring only authorized users can make modifications.
- Data Privacy & Security: User-generated content, including mind maps, quiz results, and personal study preferences, should be stored securely and encrypted to prevent unauthorized access.
- Export &Saving: Users should have the option to export mind maps as images and saving.
- AI-Powered Recommendations: The system should recommend additional learning materials based on users' mind maps, quiz results, and study behavior.
- Real-Time Feedback & Suggestions: The system should provide instant feedback when generating or modifying mind maps, ensuring users receive accurate and relevant information.

3.3.4. System Requirements

3.3.4.1. Backend Requirements

- The backend is implemented using Node.js, ensuring high performance and scalability.
- It follows a microservices architecture to enable modularity, flexibility, and ease of maintenance.
- The system provides a RESTful API for seamless interaction between the frontend and backend components.
- MongoDB is used as the primary database to store user-generated mind maps, quiz data, and learning preferences.
- LLMs such as GPT-based models are integrated to analyze user quiz performance and generate personalized mind maps dynamically.
- RAG is utilized to fetch relevant external resources for mind map enrichment.
- The system implements Google Search APIs and Adobe PDF extraction tools to collect external resources for content enhancement.

3.3.4.2. Frontend Requirements

- The frontend is developed using Angular, ensuring a responsive and interactive user experience.
- Users can generate mind maps dynamically based on their input prompts.
- The UI allows users to view, update mind maps with a user-friendly interface.
- Customizable option needed for mind maps Zoom In/Zoom out and saving.
- Mermaid.js is used for rendering mind maps in a structured and visually appealing format.

3.3.4.3. Deployment & Cloud Infrastructure

- The system is deployed using AWS for scalability and reliability.
- Docker is used for containerization, ensuring seamless deployment and environment consistency.
- A load balancer is implemented to manage traffic efficiently and maintain performance under high user loads.
- CI/CD pipelines are set up for automated testing, deployment, and version control.

3.3.4.4. Security & Performance

- User authentication is managed with JWT (JSON Web Tokens) for secure access control.
- Data encryption is implemented to protect sensitive user information and stored mind maps.
- The system ensures 99% uptime through failover mechanisms and redundancy strategies.
- API rate limiting and access control policies are enforced to prevent unauthorized access and system abuse.

3.3.4.5. Personnel Requirements

- Faculty of Computing, Sri Lanka Institute of Information Technology (SLIIT)
- University Students from Multiple Sri Lankan Universities (Survey Participants)
- Research Supervisor – Dr. Kalpani Manathunga (Assistant Professor at SLIIT)
- AI & NLP Experts for System Optimization
- Software Engineers & UX Designers for System Development

3.4. Feasibility Study

The technical, financial, and schedule considerations that were part of the study are the facts we used to determine whether the integration of an AI-driven adaptive mind map generator into a learning management system (LMS) was an appropriate and effective solution. Technical feasibility involved examining whether current technologies such as LLMs, RAG, prompt engineering, mind map visualization libraries, and microservices architecture could be successfully utilized with the existing research infrastructure. Performance metrics, system interoperability, and DevOps practices such as Docker containerization and cloud deployment were key aspects of this evaluation. Financial feasibility covered infrastructure costs, third-party API usage, and computing resource expenses. It also included cost planning for data handling, cloud storage, and integration tools. Schedule feasibility focused on defining milestone timelines, resource allocation, and risk mitigation strategies in the event of delays. The feasibility study thus covered all technical, economic, and time-based constraints, establishing a realistic and structured path toward successful development and deployment of a learner-adaptive AI-based system.

3.4.1. Technical Feasibility

Several key factors were considered to assess the possibility of implementing an AI-based mind map generator within a personalized LMS framework. These factors included infrastructure readiness, API integration capabilities, LLM compatibility, cloud-based deployment, and the availability of necessary tools and datasets. The existing computing infrastructure was analyzed in terms of processing speed, memory capacity, and storage support for data-intensive operations. Emphasis was placed on

integrating tools for document parsing (e.g., PDF extractors), mind map rendering (Mermaid.js), and performance tuning of AI models to ensure system responsiveness. Moreover, backend services were designed using a Node.js and microservices-based architecture to ensure scalability and modular integration with the Angular-based frontend. Access to frameworks, cloud APIs, and scalable data pipelines also contributed to determining the viability of the system from a technical standpoint.

3.4.1.1. Use of Large Language Models (LLMs)

One critical component in the feasibility assessment was the integration of large language models for mind map generation, document understanding, and quiz-based personalization. The evaluation focused on the memory requirements, API limits, cost efficiency, and latency of LLM inference, especially when dealing with real-time student queries or document analysis. The technical requirements of prompt engineering and fine-tuning were also reviewed to assess adaptability to varying subject domains. The availability of robust AI APIs such as OpenAI's GPT, as well as fallback mechanisms in case of API downtime, played a major role in ensuring technical reliability. The design also incorporated mechanisms for input validation, error handling, and retry strategies to enhance model robustness. Performance indicators such as relevance of content, semantic coherence in mind maps, and the accuracy of weak-point detection were essential in this evaluation.

3.4.1.2. Use of Retrieval-Augmented Generation (RAG)

During the evaluation of the technical feasibility for applying RAG, several key factors were identified to validate its applicability in an adaptive learning environment. Primarily, the infrastructure's compatibility with RAG architecture was assessed, focusing on the integration of retrieval systems with LLMs to enhance context relevance and minimize hallucination. The system required access to reliable document storage and fast semantic search capabilities for efficient retrieval. Considerations such as document preprocessing, vector embedding generation, and query indexing were evaluated to ensure that relevant knowledge was consistently fetched during prompt execution. The feasibility analysis also reviewed the reliability

of open-source retrieval frameworks and API-based solutions in terms of performance, cost, and deployment scalability. Attention was paid to latency, relevance of retrieved content, and the ability to serve diverse topics within real-time interactions. Overall, the incorporation of RAG was deemed technically viable, enabling the generation of more grounded and user-personalized outputs in the LMS.

3.4.1.3. Use of Prompt Engineering & Mind Map Visualization Tools

Another core component in the feasibility study was the use of prompt engineering in combination with mind map visualization frameworks such as Mermaid.js. The evaluation explored how precisely crafted prompts could be used to guide the output of large language models toward generating clear and semantically structured mind maps. The effectiveness of prompt engineering was assessed by testing its adaptability across multiple learning domains and evaluating consistency in the structure of output diagrams. Furthermore, the feasibility of rendering these mind maps dynamically in the frontend using tools like Mermaid.js was studied, including aspects like browser compatibility, rendering time, and the customization of visual styles. This component also required reviewing user interaction capabilities, such as saving, editing, and replacing existing maps, which was supported by local storage or database integration. From a performance perspective, the impact of prompt length, LLM response time, and visual rendering accuracy were critical indicators. The analysis confirmed that these tools collectively supported the system's personalization goals, making them a feasible and effective part of the technical design.

3.4.1.4. Knowledge of Deployment and Integration

The deployment strategy for the adaptive mind map generator was designed to support scalability, modularity, and continuous integration. The backend, built with Node.js, and the frontend in Angular were containerized using Docker. These containers were orchestrated and managed via AWS services and automated through GitHub Actions for CI/CD. MongoDB Compass was employed to store and retrieve structured data including prompts, mind maps, and quiz performance metrics. RESTful APIs facilitated seamless communication between the LLM engine and LMS application.

The system’s architecture allowed for modular upgrades and easy debugging, ensuring a robust deployment environment that supports long-term expansion and real-world usage.

3.4.1.5. Knowledge of Resource Generation using APIs and AI Services

An essential component of the adaptive mind map system was the integration of external knowledge sources through automated resource generation, using a variety of AI services and APIs. The objective was to enhance learning accuracy and context relevance by pulling supplementary content dynamically from reliable sources and generative models.

Google Search API and Bing Search API were incorporated to retrieve high-quality web resources, such as academic articles, blog posts, and verified tutorials, related to the user’s quiz topics and identified weak areas. These APIs were queried using dynamically crafted prompts based on user input, and the responses were filtered through predefined relevance criteria such as domain authority, recency, and content structure.

In parallel, GPT-based models were used to refine and summarize the retrieved content, ensuring that the resulting text aligned with the user's learning style and the structural requirements for mind map generation. The GPT engine also provided definitions, breakdowns, and conceptual explanations of topics where direct search content was lacking.

To further enhance visual engagement and understanding, DALL·E was integrated for generating contextual images related to mind map nodes. These visuals served to anchor knowledge and improve memory retention, especially for visual learners. The images generated were based on descriptions dynamically derived from the node content, and were optimized to maintain clarity and thematic alignment.

The technical feasibility assessment for this component involved ensuring:

- API rate limits were respected and efficiently managed.

- Responses were processed and validated for accuracy and relevance.
- The integration with LLMs maintained a low-latency and seamless user experience.
- The system could dynamically assign these generated resources to appropriate nodes within the mind map without affecting performance or user interactivity.

This hybrid resource generation strategy proved to be highly feasible and effective, leveraging both search-based and generative approaches to offer a comprehensive, up-to-date, and deeply personalized learning environment.

3.4.2. Schedule Feasibility

The schedule feasibility of the adaptive learner-centric mind map generation system was evaluated by conducting a thorough analysis of the project lifecycle to ensure timely and successful delivery. A detailed project timeline was developed, segmented into key phases including literature review, requirement elicitation, system architecture planning, backend and frontend development, AI service integration, testing, and final deployment.

The project began with an extensive literature review and requirement analysis to align with the current trends in personalized education and adaptive learning. This phase was followed by a design period where the architecture of the LMS was structured using microservices, defining the interactions between the Node.js backend, Angular frontend, MongoDB database, and external AI services such as LLMs, Azure Custom Vision, GPT.

Development was divided into iterative sprints, each focused-on core components such as quiz result processing, adaptive logic for mind map generation, visualization rendering using Mermaid.js, integration with APIs (Google Search, Bing Search, DALL·E), and deployment configurations on Docker with GitHub Actions and AWS. Each sprint included designated milestones to ensure progress tracking and issue resolution in real-time.

Resource dependencies such as access to AI APIs, storage services, and deployment infrastructure were identified early and resolved through pre-configuration and automation. A Gantt chart was used to map dependencies and conduct critical path analysis, helping prioritize high-impact tasks such as LLM integration and model inferencing for mind map updates.

Risk assessments were conducted at each stage to account for potential delays, such as unresponsive API services, model integration conflicts, or frontend-backend mismatches. Contingency strategies were implemented including fallback APIs, mock data testing, and modular component design to allow parallel development.

To ensure schedule adherence, weekly reviews and progress meetings were held with the research and development team. Resource allocation was dynamically adjusted to critical components when delays were predicted, and redundant development paths were established to avoid bottlenecks.

Overall, the scheduling plan provided a well-balanced timeline that maintained productivity, accommodated unforeseen events, and guaranteed the project met its milestones within the planned academic term. The system was successfully implemented on time with all core features adaptive quiz analysis, mind map generation, resource integration, and interactive visualizations fully functional and aligned with the defined objectives

3.4.3. Economic Feasibility

Prior to the development of the Adaptive Learner-Centric Mind Map Generation System, a detailed economic feasibility study was conducted to evaluate the potential financial implications and overall return on investment. The cost analysis covered various domains including infrastructure setup, software services, API integration, development resources, and long-term maintenance.

The primary expenditure areas included backend development using Node.js, frontend implementation in Angular, and database management via MongoDB. Additionally, costs were allocated for deploying the system in a containerized Docker environment,

leveraging cloud services like AWS, and automating workflows through GitHub Actions. The integration of advanced AI tools including GPT-based LLMs, Google Search API, Bing Search API, and DALL·E for resource generation was examined for subscription models and pay-per-request pricing structures.

Further, the investment in visualization tools such as Mermaid.js was evaluated, including efforts spent to customize and dynamically render interactive mind maps. Cost considerations were also made for training and documentation resources to ensure usability and maintainability of the platform in educational institutions.

The project team analyzed the human resource expenditure, involving roles such as AI specialists, web developers, UI/UX designers, and researchers contributing to adaptive logic, knowledge extraction, and frontend rendering. The effective allocation of human capital was prioritized to minimize overhead and maximize productivity within the academic term's timeframe.

A comprehensive cost-benefit analysis was performed by weighing the predicted expenses against both tangible and intangible benefits. Tangible benefits included the potential for the system to reduce manual teaching workload, improve student performance, and personalize content delivery. Intangible benefits consisted of enhanced student engagement, increased retention, and the long-term scalability of adaptive learning models within digital education platforms.

The prospective ROI was evaluated not just from a financial standpoint, but also in terms of academic value and institutional adoption. Open-source technologies and academic license plans were favored wherever possible to reduce costs. Moreover, availability and ease of access to APIs and cloud credits under educational programs contributed to economic viability.

This financial analysis confirmed that the adaptive learning system, with its integration of advanced AI, resource generation capabilities, and personalized visualization, could be developed and deployed within budget constraints while delivering significant educational impact and scalability in the long run.

3.4.4. Financial Feasibility

- The required resources, including software tools, libraries, and cloud services, are cost-effective and fit within the project budget.
- AWS offers cost-effective cloud solutions suitable for this system's scale, and the use of Docker ensures a consistent deployment environment with minimal infrastructure overhead.
- The project does not require expensive hardware or specialized equipment, and most of the development tools are open-source or have free-tier options available.
- The project can be maintained with minimal costs after deployment, as AWS scaling can be adjusted based on usage.

3.4.5. Market Feasibility

- The adaptive mind map generation system is highly relevant to the educational technology market, specifically in areas such as personalized learning and AI-driven educational tools**.**
- The system offers an innovative approach to learning, where mind maps can be automatically tailored to students' needs based on their quiz performance and cognitive preferences.
- The growing interest in AI and adaptive learning systems positions the system to compete well in the market. There is a demand for personalized learning tools, and this project aims to address that gap.
- A comprehensive market analysis shows that the demand for AI-driven educational tools is increasing, with competitors such as Quizlet and MindMeister. However, our unique focus on personalized mind map generation and integration with LLMs gives us a competitive edge.
- The market potential for this system is substantial, and its ability to enhance learning outcomes gives it a strong value proposition for educational institutions and learners.

3.4.6. Operational Feasibility

- The team is well-equipped to manage the entire software development lifecycle, from requirement analysis to deployment and maintenance.
- The system's modular design ensures that the development process is well-structured, and each component can be iteratively improved.
- The system meets the needs of the users by providing personalized mind maps, and multimedia integration, all of which contribute to a rich learning experience.
- The user interface is designed to be intuitive, making it easy for students and educators to interact with the system and generate mind maps without significant training.
- The product’s scalability ensures that it can be easily adopted by a large number of users, and its flexibility allows for future enhancements as the needs of the users evolve.

3.4.7. Ethical Feasibility

- Ethical considerations regarding data privacy and security are addressed by implementing strong encryption mechanisms for storing and transferring user data.
- The system adheres to data protection regulations and ensures that the user data is confidential, following best practices for handling personal information.
- The system will not collect any sensitive data beyond the user's quiz results, mind maps, and learning preferences, ensuring that the research and development process complies with ethical standards.

3.5. Project breakdown

3.5.1. Gantt Chart

No	Assessment / Milestione	2024- 2025													
		April	May	June	July	August	September	October	November	December	January	February	March	April	May
1	Research group formation														
2	Supervisor selection														
3	Brainstorming workshop 1														
4	Selection of research topic														
5	Co-supervisor selection														
6	Brainstorming workshop 2														
7	Feasibility and background study														
8	Topic registration from submission														
9	In-depth feasibility and background study 1														
10	External supervisor selection														
11	Topic assessment form submission														
12	Topic assessment from evaluation														
13	In-depth feasibility and background study 2														
15	Proposal presentation														
16	Individual proposal report submission														
17	Implementation of research work(upto 50%)														
18	Progress presentation 1														
19	Prepare and submit research paper														
20	Implementation of research work(upto 90%)														
21	Progress presentation 2														
22	Integration of the research work														
23	Project completion														
24	System testing														
25	Website and final report preparation														
26	Final presentation														
27	Final report submission														

Figure 1: Gatte Chart

3.5.2. Work Breakdown Structure

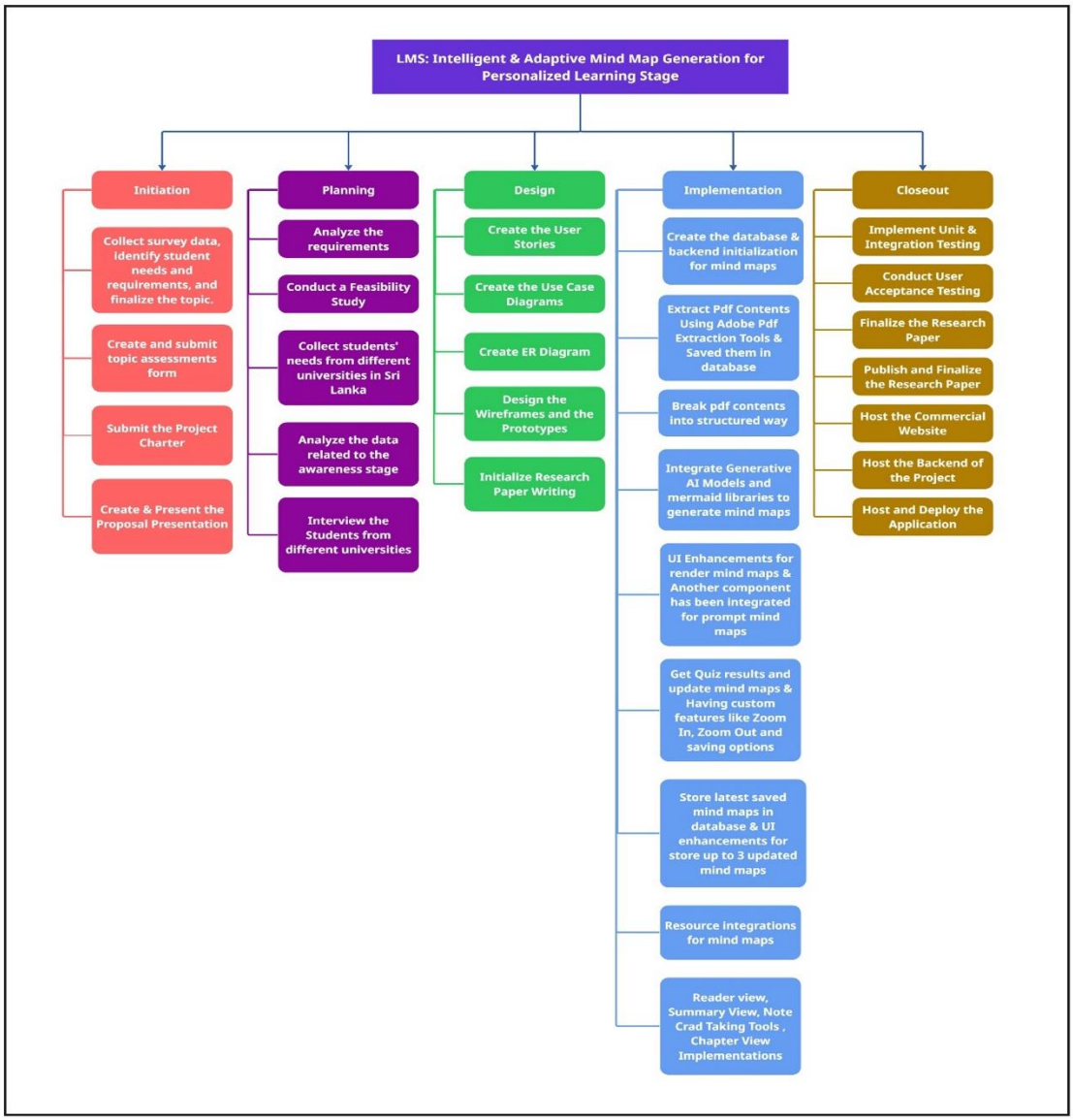


Figure 2: Work Breakdown Structure

3.5.3. Design Diagram

3.5.3.1. Use Case Diagram

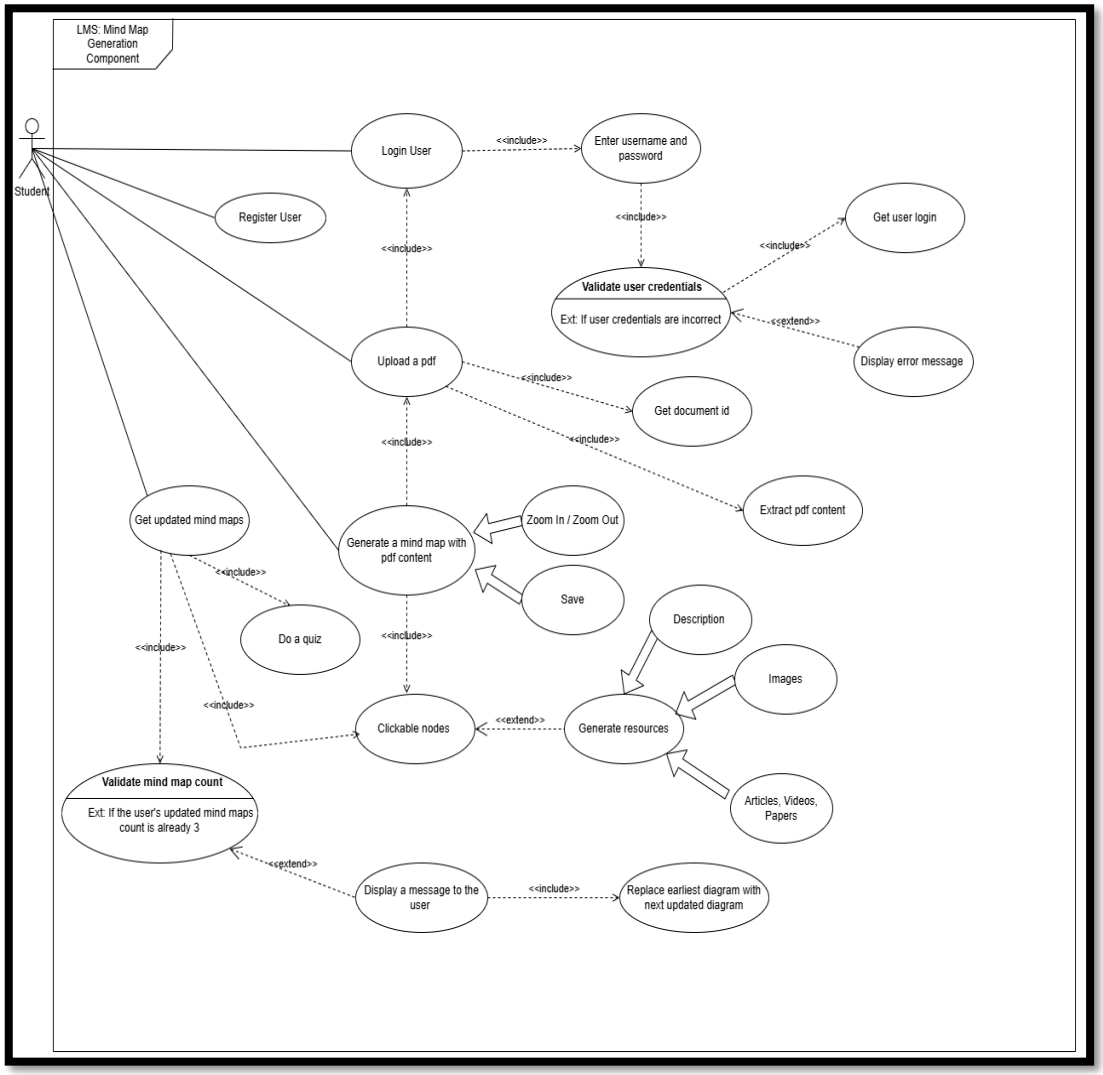


Figure 3: Mind Map generation Use case

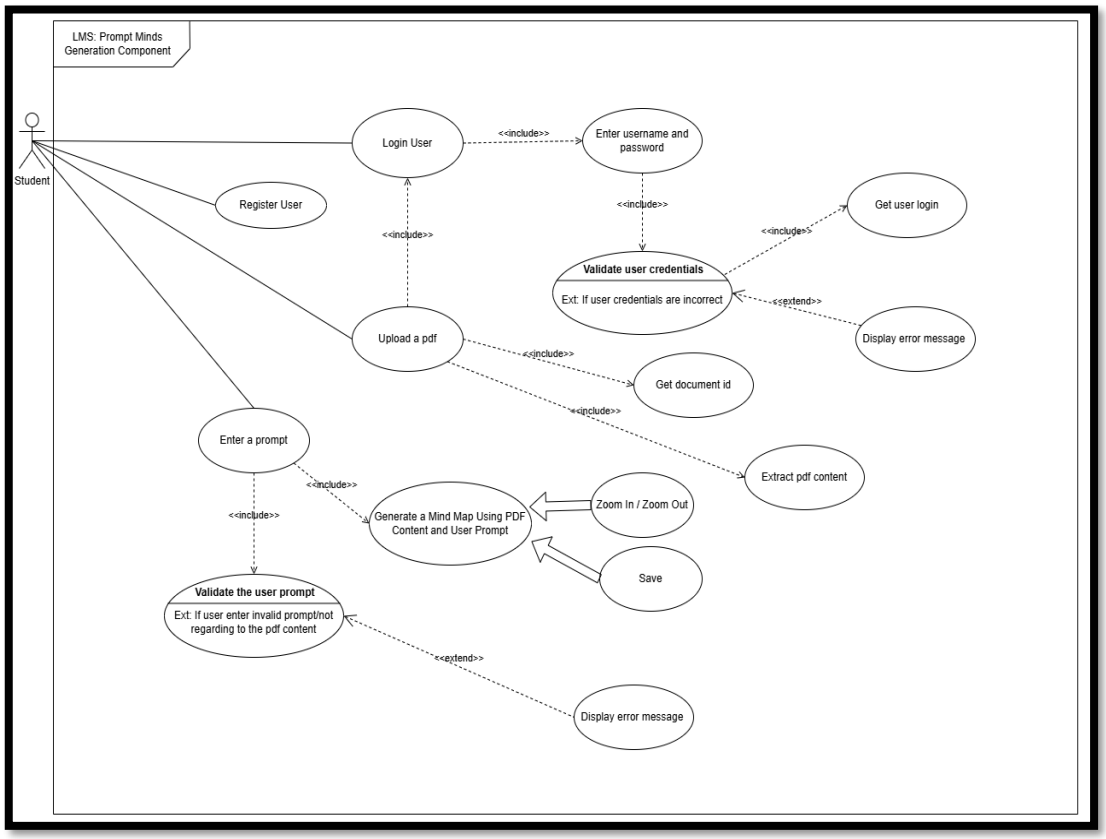


Figure 4: Prompt Mind Use Case

3.5.3.2. Sequence Diagram

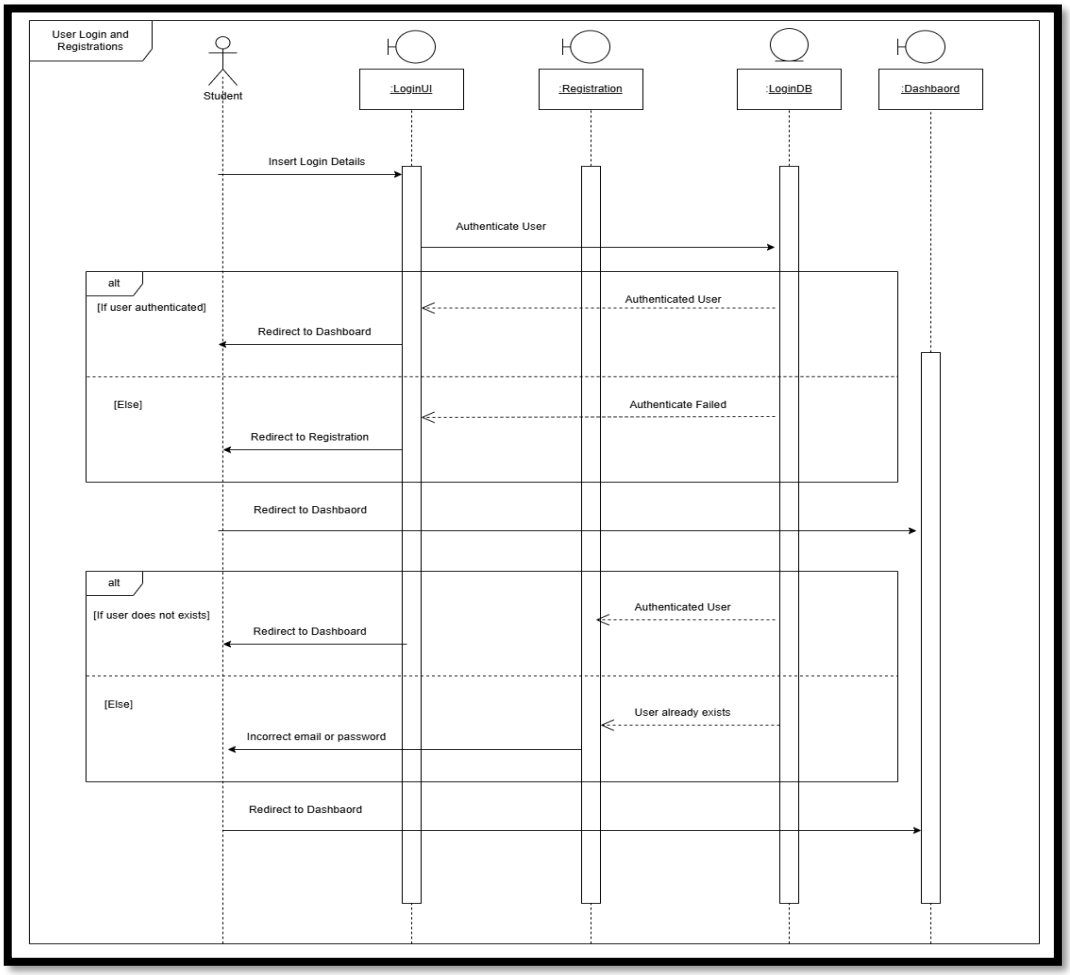


Figure 5:Sequene Diagram

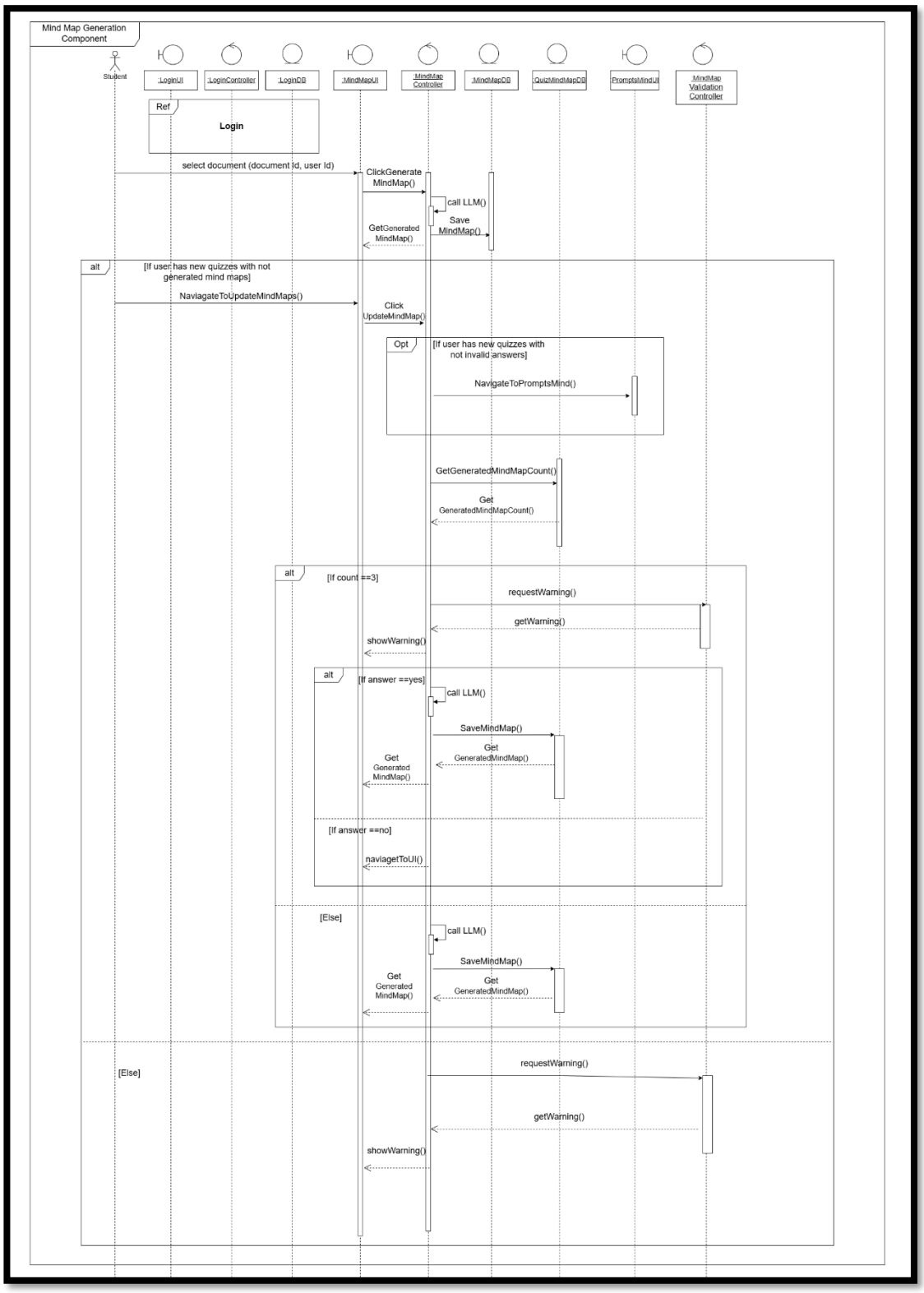


Figure 6: Mind Map Genaration Sequence diagram

3.6. Development Process

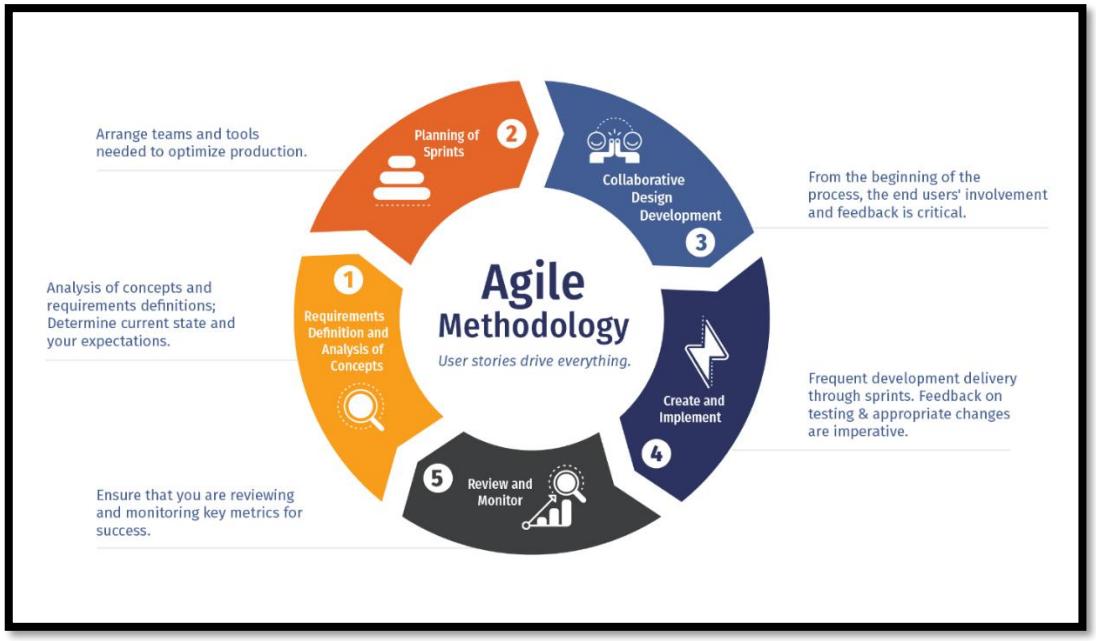


Figure 7: Agile Development Process

Considering all the other processes available, it was without doubt that Agile Methodology was most likely to be chosen as following it gives much flexibility compared to the other alternatives. In the process of developing the Adaptive Mind Maps, the use of Daily Scrums and Kanban board utilization will be considered as 15 these characteristics increase collaboration amongst the team members and boost development process.

3.7. System Architecture

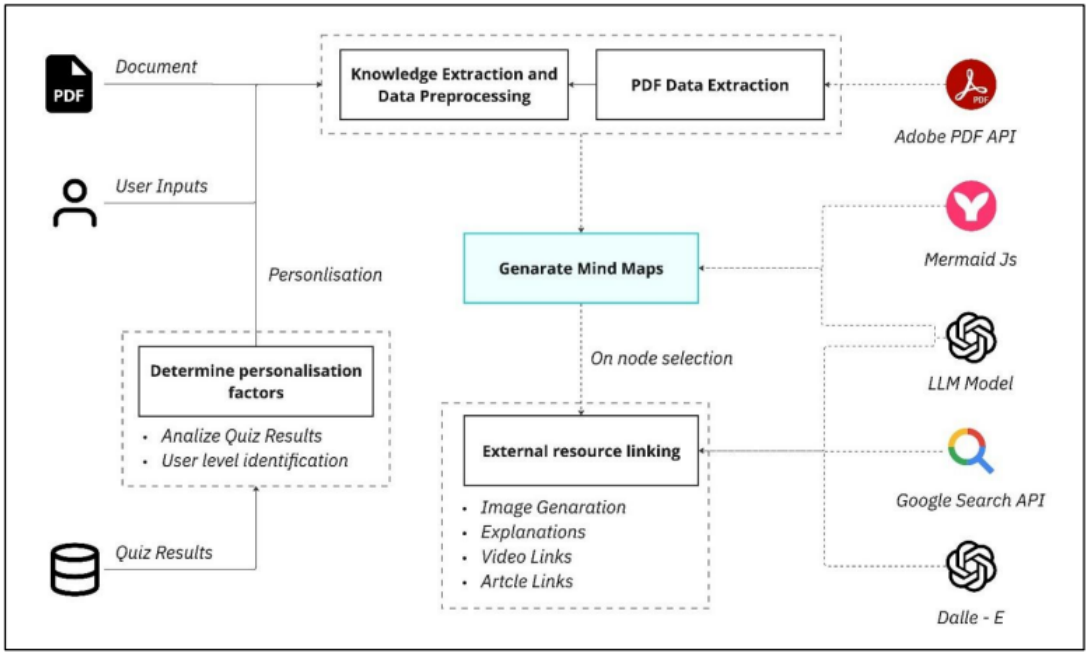


Figure 8: Component Diagram Mind Map

The proposed system architecture integrates multiple advanced technologies to create a seamless and efficient platform for adaptive learning and mind map generation. The architecture is designed to handle the complexity of personalized learning by combining AI-driven content analysis, dynamic mind map generation, and interactive visualizations within a unified system.

At the core of the system, user data specifically quiz results and performance metrics are collected through the LMS interface, built with Angular. This data is then processed and analyzed to identify the learner's weak areas. The backend, developed using Node.js, serves as the communication bridge between the frontend interface and the underlying technologies. MongoDB stores user-related data, including quiz results, mind maps, and user-specific prompts, ensuring data persistence and easy retrieval.

LLMs such as GPT are integrated into the system to assess quiz performance and provide personalized learning suggestions. The LLMs also play a pivotal role in generating dynamic prompts that help guide the learner's study approach. These AI-driven prompts are leveraged to generate personalized mind maps that target specific areas where the learner needs improvement.

The Mermaid.js library is employed to generate interactive, hierarchical mind maps, which are rendered dynamically on the frontend. These mind maps can be customized by the user and are updated based on their ongoing quiz results. The system allows up to three stored mind maps per user, automatically replacing the oldest map as new ones are generated.

In terms of resource generation, the system integrates Google Search API**,** Bing Search API, and DALL·E to dynamically fetch relevant resources and create customized visual aids. These resources are seamlessly embedded into the mind maps, enriching the learning content and further personalizing the educational experience.

The deployment of the system is managed within Dockerized environments using AWS, ensuring scalability, security, and efficient resource management. Continuous integration and deployment are handled through GitHub Actions, allowing for automated testing, builds, and updates.

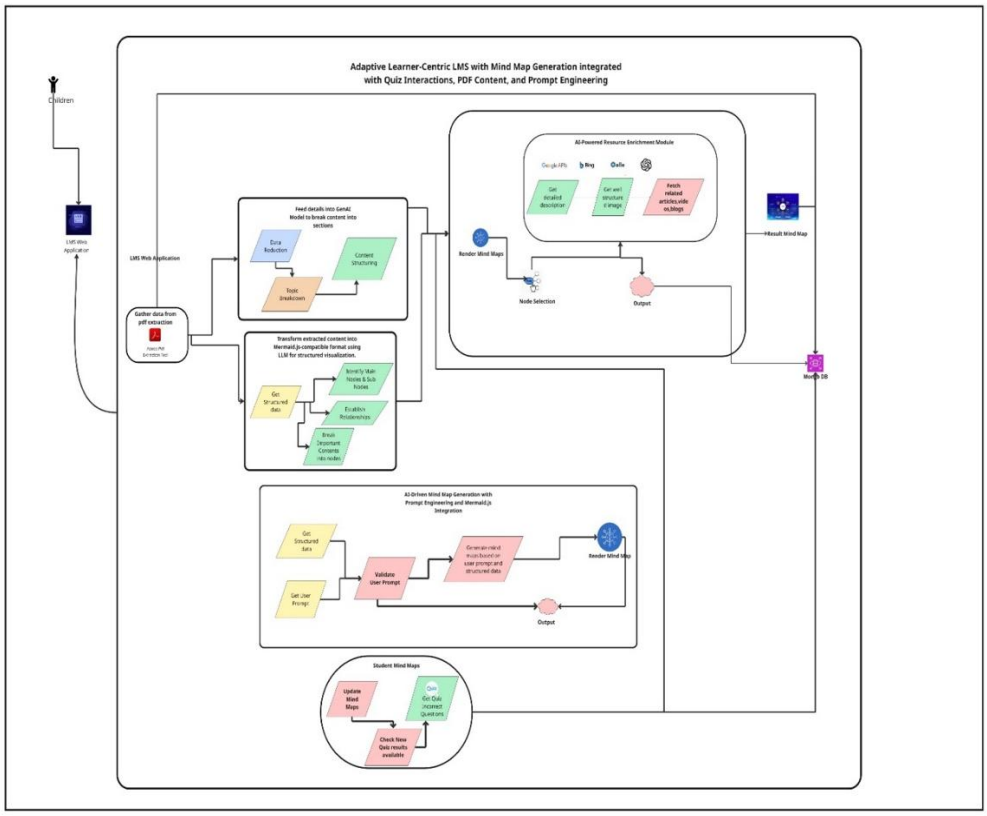


Figure 9: Mind Map Genaration Component Diagram

As depicted in the component diagram, the Adaptive Learner-Centric LMS is designed to integrate mind map generation with quiz interactions, PDF content extraction, and prompt engineering. The system is structured to ensure seamless data flow between different components, enabling an AI-driven approach to personalized learning. The LMS web application acts as the primary interface where students can upload study materials in PDF format or input custom prompts to generate mind maps. The system then processes the extracted data using LLMs such as GPT-4 and GPT-4o-Mini, ensuring structured content breakdown and effective visualization.

The processing begins with data extraction from PDFs, where the system reduces unnecessary information and categorizes content into structured sections. This extracted data is then transformed into a format compatible with the mind map generation module, which involves identifying main nodes and sub-nodes, establishing relationships between concepts, and breaking down important information into smaller, digestible components. Once the structured content is ready, the AI-driven mind map generation module takes over, using prompt engineering techniques to refine user inputs and ensure the most relevant output. The mind maps are then visually represented using Mermaid.js, allowing for a structured and intuitive learning experience.

To enhance the accuracy and depth of mind maps, an AI-powered resource enrichment module is integrated into the system. This module leverages Google APIs, Bing search, and DALL·E to fetch external resources such as well-structured descriptions, additional articles, videos, and even AI-generated images that support the learning process. By incorporating these real-time knowledge sources, the LMS ensures that students receive up-to-date, well-organized study materials that go beyond static content. The resource enrichment module further refines the output by verifying the accuracy of the generated content, helping students gain deeper insights into complex topics.

A key feature of this system is its adaptive learning mechanism, where mind maps evolve based on student interactions and quiz performance. After completing a quiz, the system analyzes the results, identifies incorrect responses, and updates the mind

map to emphasize weak areas that require further attention. The LMS continuously monitors quiz results to check for updates, ensuring that students have an evolving study resource tailored to their learning progress. Additionally, to maintain an efficient storage mechanism, students can store up to three mind maps, with the earliest one being replaced when a new mind map is generated. This approach keeps the learning process dynamic while preventing excessive data accumulation.

Finally, all generated mind maps and quiz data are securely stored in the Mind Map Database, allowing students to retrieve and review their past learning materials when needed. The structured workflow ensures that every component function cohesively, from data extraction and processing to AI-enhanced visualization and continuous adaptation based on user interactions. Through this system, students receive an AI-driven, personalized, and interactive learning experience, optimizing their study efficiency with real-time resource enrichment and structured knowledge visualization.

3.7.1. Backend Integration

The integration of the backend system is essential for supporting the adaptive learning workflow, ensuring secure, efficient, and dynamic generation and management of personalized mind maps. The backend is primarily responsible for handling data processing, user authentication, structured content management, and persistent storage of generated mind maps and user interactions.

The process begins with the input of PDF content, which is handled through Adobe PDF extraction APIs. This raw content is processed by LLMs to extract and convert the data into a structured format. The structured content, along with associated user details and a unique document ID, is stored in the database. This structured data is later utilized for downstream processes such as pdf-mind map generation and quiz-based mind map generation.

Once the document is processed, the structured content is forwarded to the frontend for visualization and mind map generation. Although the frontend handles the rendering, the backend includes key functionalities for storing and managing the

generated mind maps. Each mind map is saved in a MongoDB database, linked to both the user ID and the corresponding document ID, ensuring traceability and personalization.

In addition to the base mind map, the system supports three adaptive mind maps, which are generated based on quiz results. These mind maps highlight weak areas in the learner's understanding and are also saved in the database under the same user-document relationship. The backend ensures that only up to three adaptive mind maps are stored per user per document; the oldest map is automatically replaced when a new one is generated.

Moreover, user interactions with mind maps are tracked and stored by the backend. When a user clicks on a specific node, this interaction is recorded in the database. Each node in a mind map also carries a status flag such as *done*, *in progress*, or *not done* which can be updated by the user. The backend manages these updates, storing both the updated status and any linked resources associated with the node, supporting a more personalized and goal-oriented learning experience.

Crucially, all of these backend functionalities are accessible only to authenticated users, ensuring secure access and user-specific data integrity. Authentication safeguards all processes including content processing, mind map generation, data updates, and storage, maintaining a secure and personalized learning environment for every student.

3.7.2. REST API for Communication between Backend and Frontend

A RESTful API serves as the vital communication channel between the Node.js backend and the Angular frontend, ensuring smooth data transmission and integration of various system components. The API facilitates seamless interaction between the user interface and the backend system, making the adaptive learning process accessible and efficient.

When a student completes a quiz, the results are sent from the frontend to the backend via the REST API. This triggers the backend system to process the quiz data, using it to assess the student's performance. Once the performance analysis is complete, the frontend generates a personalized prompt. This prompt is used by LLMs to generate adaptive mind maps tailored to the student's weak areas.

The REST API also handles the retrieval of mind maps**,** which are then displayed to the user in an interactive format. It ensures that up-to-date mind maps are always presented to the student based on their latest quiz performance. Additionally, the API supports the storage and retrieval of user data, such as quiz results, stored mind maps, and user preferences, which are all stored in the MongoDB database**.**

By using the REST API, the system ensures that the frontend remains lightweight and responsive, providing a smooth user experience while handling complex data processing tasks in the backend. This API-driven communication is crucial for delivering dynamic and personalized learning experiences to students.

3.7.3. System Deployment

The deployment of the Adaptive Learner-Centric Mind Map Generation System is carried out using modern cloud technologies to ensure scalability, reliability, and ease of maintenance. The backend system**,** developed with Node.js**,** is deployed in Dockerized environments on AWS for optimal performance, security, and scalability. This cloud deployment ensures that the system can handle an increasing number of users and large amounts of data without compromising performance.

The frontend, developed with Angular, is integrated seamlessly with the backend through the RESTful API. The system is designed to be fully responsive, providing students with access to their personalized mind maps on various devices, including desktops and tablets.

Additionally, the entire deployment process is automated using GitHub Actions, ensuring CI/CD. This setup allows for regular updates to be deployed automatically, ensuring that the system remains up-to-date with the latest features and improvements.

Overall, the deployment strategy ensures that the system is highly scalable, secure, and reliable, providing an optimal learning experience for students.

3.7.4. System Solution

The solution for the Adaptive Learner-Centric Mind Map Generation System is built around the core objective of providing personalized learning experiences to students based on their quiz performance. This solution encompasses a web application that allows users to upload their quiz results, which are then analyzed by the backend to generate adaptive mind maps.

The web interface offers a simple, user-friendly experience for students, enabling them to easily access their performance data and personalized mind maps. The system ensures that mind maps are dynamically generated based on individual learning needs, with a focus on areas where students have demonstrated weaknesses. These personalized mind maps guide students through their learning journey, helping them focus on the concepts that require more attention.

Students can interact with the system through the web interface, which is designed to be responsive and accessible on various devices, from desktops to tablets. The system’s integration with LLMs**,** such as GPT, ensures that the mind maps are rich with relevant learning content. Through continuous analysis and integration of the student's quiz performance data, the system continuously refines and updates the mind maps, keeping them in sync with the student's evolving learning needs.

The solution's flexibility allows it to cater to various educational contexts, enhancing student engagement and learning efficiency. It empowers learners by providing a dynamic, adaptive learning environment that responds to their individual progress and needs.

3.7.5. Software Design

The Adaptive Learner-Centric Mind Map Generation System leverages a combination of cutting-edge technologies, including Node.js**,** Angular**,** LLMs**,** Mermaid.js, and MongoDB, to provide an efficient and dynamic learning experience.

The backend is built with Node.js**,** which powers the entire system and handles data processing tasks, such as analyzing quiz results and generating personalized prompts. These prompts are processed by LLMs (such as GPT models) to generate detailed mind maps that focus on areas where the student has shown weaknesses. The RESTful API serves as the communication bridge between the backend and the frontend, ensuring seamless interaction between the system's various components.

For the frontend, Angular is used to create a responsive and interactive web interface, allowing students to easily upload their quiz results and view the generated mind maps. The Mermaid.js library is integrated to dynamically generate interactive mind maps that visually represent the key learning concepts. These mind maps are updated in real-time, based on the latest performance data, ensuring that students always receive the most relevant and up-to-date learning resources.

The data architecture is built using MongoDB, which is a NoSQL database that allows for flexible storage of user data, quiz results, and mind maps. MongoDB's schema-free approach ensures that the system can adapt to changing data needs as the project evolves. It provides fast and reliable access to the stored data, enabling efficient retrieval and updates of mind maps and quiz results. MongoDB's distributed network ensures that the system can handle a growing amount of data without experiencing slowdowns, offering scalability and high performance. This architecture provides a solid foundation for the continuous development of the system while ensuring secure and reliable data management.

The entire software design ensures that the system is highly adaptable, scalable, and user-friendly, offering an optimal learning experience that grows and evolves with the student's learning journey.

3.8. Researched Domain

3.8.1. Mind Map Generation

Mind map generation is an advanced visualization technique used for organizing, structuring, and representing knowledge dynamically. In an AI-driven LMS, mind maps act as personalized study tools that enhance comprehension by visually linking related concepts.

AI-based mind map generation utilizes techniques such as RAG**,** LLMs, and prompt engineering techniques to generate structured diagrams dynamically based on user input, quiz performance, and external resource integration. Mind maps are generated through intelligent content extraction, relationship mapping, and personalized adaptation.

Several approaches can be taken for automated mind map generation, depending on system requirements and available technologies. Common methods include:

- Rule-Based Mapping: Uses predefined logic and structured keyword extraction to create node connections.
- AI-Powered Generation**:** Uses LLMs like GPT-4, DALL-E, Gemini, or Claude to generate contextualized mind maps dynamically.
- Hybrid Mapping**:** Combines AI-generated insights with user-defined modifications for a semi-automated experience.

The decision on the mind map generation method is based on project requirements, computational constraints, and user adaptability.

3.8.2. Adaptive Mind Map Personalization

Adaptive mind map personalization involves dynamically updating the generated mind maps based on user progress, quiz results, and external knowledge sources. This

process ensures that the mind maps evolve alongside the learner's understanding. Key AI-driven techniques used in mind map adaptation include:

- Quiz-Driven Personalization: The system updates mind maps based on the learner's quiz performance, reinforcing weak concepts and adjusting focus areas accordingly.
- RAG: Enhances the mind map by integrating real-time external information, ensuring accurate and up-to-date learning content.
- Progressive Mind Map Evolution: Users can store up to three mind maps, and when a new map is created after a quiz, the oldest stored mind map is replaced to maintain relevancy.
- External Resource Integration: AI-driven APIs, such as Google Search API, Bing Search API, fetch additional study materials to enrich the mind map content dynamically.

By leveraging LLMs, external resource retrieval, and user behavior analysis**,** the system optimizes mind map generation, ensuring a highly personalized and effective study experience.

3.8.3. Mind Map Generation Techniques

Several mind map generation techniques can be employed to refine the quality of AI-generated mind maps. Some of the widely used techniques include:

- Hierarchical Mind Mapping: A structured approach where concepts are arranged in a tree-like structure with a central node and branching sub-nodes.
- Graph-Based Mind Mapping: Utilizes AI to create complex, interconnected relationships between multiple concepts dynamically.
- Conditional Node Generation: Adapts the mind map structure based on predefined conditions, such as user preferences or quiz results.
- Prompt Engineering Techniques for Mind Map Generation: Prompt engineering techniques are strategically applied in this system to guide large language models (LLMs) for accurate and context-aware mind map generation. The approach

utilizes well-established methods to ensure structured outputs and meaningful insights:

- Instruction-Based Prompting is primarily utilized in PDF-based mind map generation, quiz-based adaptive mind maps, and prompt-based mind map creation. This technique involves giving explicit instructions to the LLM, such as breaking content into hierarchical structures using Mermaid.js syntax or focusing on incorrect quiz responses for remediation. This ensures consistent and targeted outputs tailored to specific user needs.
- Few-shot Prompting is applied in PDF-based mind map generation and quiz-based adaptive mind maps. A few examples of correctly generated mind map structures or question-response pairs are included to guide the model. This helps the LLM understand the expected format and logic, improving relevance and coherence in mind map generation.
- Zero-shot Prompting is leveraged in resource-enhanced mind map construction and description generation, where no prior examples are provided. The model responds to novel queries such as questions related to the content by attempting to extract relevant concepts or flagging them as invalid if outside the document scope.
- Role Prompting is employed in resource-enhanced mind map construction and description generation, where the model is assigned specific roles (e.g., subject expert, assistant educator) to improve contextual relevance. This improves the quality of explanations, resource suggestions, and narrative descriptions linked to mind map nodes.
- Chain of Thought Prompting is implemented for question validation prompts. The model is guided to reason through a series of logical steps to determine whether a user's question is relevant to the PDF content. This multi-step reasoning ensures accurate content filtering and feedback.

This integrated prompting strategy ensures that mind maps are not only accurate and well-structured, but also adaptive to users' learning needs and personalized learning paths.

3.8.4. Large Language Models (LLMs)

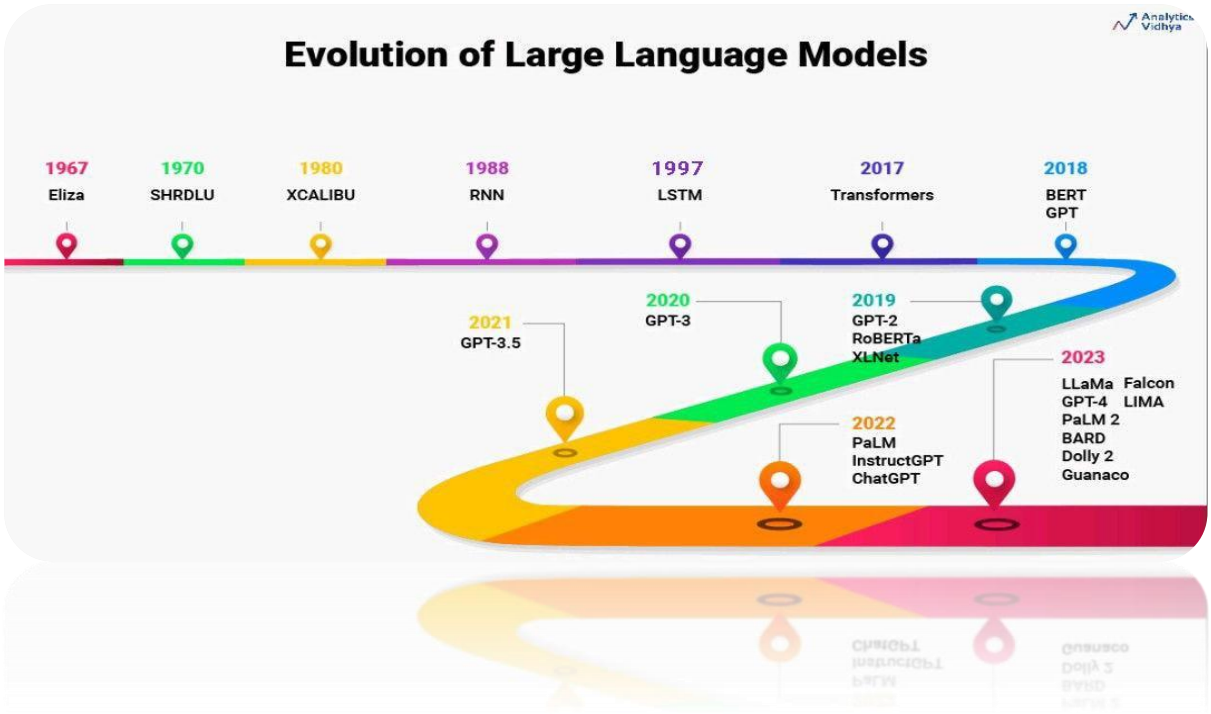


Figure 10: Evolution of Large Language Models

Large Language Models (LLMs) are advanced artificial intelligence models capable of processing and generating human-like text based on extensive training on diverse datasets. These models leverage deep learning architectures, particularly transformers, to understand context, generate responses, and assist in various applications without requiring task-specific training.

In the context of AI-driven mind map generation, LLMs such as GPT-4, GPT-4o-Mini, and DALL·E play a crucial role in dynamically creating, structuring, and refining mind maps based on user input, quiz performance, and external knowledge sources.

3.8.5. Applications of LLMs in Mind Map Generation

LLMs enhance mind map generation by:

- Content Extraction & Structuring: Identifying key concepts from user input, documents, or external sources and organizing them into hierarchical mind maps.
- Adaptive Mind Map Evolution: Refining mind maps dynamically based on quiz results, user performance, and learning patterns.
- Contextual Understanding & Expansion: Generating meaningful relationships between concepts, providing explanations, and suggesting additional related topics.
- Visualization Assistance: Supporting DALL·E for image-based mind maps, enhancing the visual representation of complex topics.

3.8.6. LLM-Based Mind Map Generation Process

1. User Input Processing

   - The user provides a topic, custom prompt, or quiz result.
   - The system analyzes the input and extracts key concepts.

2. Concept Structuring & Hierarchical Mapping

   - LLMs (e.g., GPT-4, GPT-4o-Mini) categorize information into nodes and sub-nodes**.**
   - Relationships between concepts are established based on context.

3. External Knowledge Integration (Retrieval-Augmented Generation - RAG)

   - The system fetches real-time information from sources using APIs (e.g., Google Search API).
   - The mind map is refined using updated and relevant data**.**

4. Visual Representation Enhancement

    - DALL·E generates graphical representations to complement textual mind maps.
    - AI ensures that illustrations align with the structured knowledge.

5. User-Based Adaptation & Evolution

    - Quiz data and user interactions modify existing mind maps.
    - If a user has three saved mind maps, the earliest one is replaced.

3.8.7. Advantages of Using LLMs for Mind Map Generation

- Efficiency & Automation: Eliminates the need for manual structuring, saving time for learners.
- Personalized Learning Experience: Tailors mind maps based on individual performance and knowledge gaps.
- Real-Time Knowledge Retrieval: Ensures content accuracy by integrating external sources dynamically.
- Multimodal Capabilities: Combines text-based mind maps with AI-generated images for better comprehension.

3.8.8. Challenges & Considerations

- Token-Based Cost: LLMs operate on a **"pay-per-token"** model, requiring budget-conscious implementation**.**
- Accuracy & Verification: Retrieved knowledge must be fact-checked to prevent misinformation.
- Computational Resources: High-performance AI models require efficient backend management to optimize response time and storage.

3.9. Implementation

This research presents a software solution that includes the development and implementation of a web-based Adaptive Learner-Centric LMS, designed to enhance the educational experience of students through personalized mind map generation. The system allows learners to visually understand and interact with educational content via dynamically generated mind maps that break down key concepts, subtopics, and their relationships.

The mind map generation component is driven by an AI model that analyzes quiz performance and learning behavior to identify students' weak points. Based on this data, the system automatically creates personalized mind maps and provides external learning resources such as summaries, links, and definitions. Students also have the option to input custom prompts, which the system uses to generate mind maps tailored to the topic of their choice.

The goal is to provide a more adaptive, visual, and efficient learning experience by integrating advanced AI features with modern web technologies. Students can explore study material, improve weak areas, and receive on-demand support all through a smart, interactive web interface.

For the implementation of the system, the following technologies were used:

- **Visual Studio Code** – for code development
- **Postman** – for testing APIs
- **MongoDB Compass** – for visualizing and managing the database
- **Node.js** – for backend development
- **Express.js** – for building RESTful APIs
- **Angular** – for frontend development
- **Docker** – for containerization of services
- **DockerHub** – for storing and managing container images
- **GitHub** – for version control
- **GitHub Actions** – for CI/CD automation

- **AWS (Amazon Web Services)** – for hosting and cloud infrastructure
- **OpenAI GPT (LLM)** – for personalized mind map and content generation
- **Mermaid.js** – for mind map and diagram visualization
- **Google Search API** – for external resource retrieval
- **PDF Extraction Tools** – for document parsing and summary generation

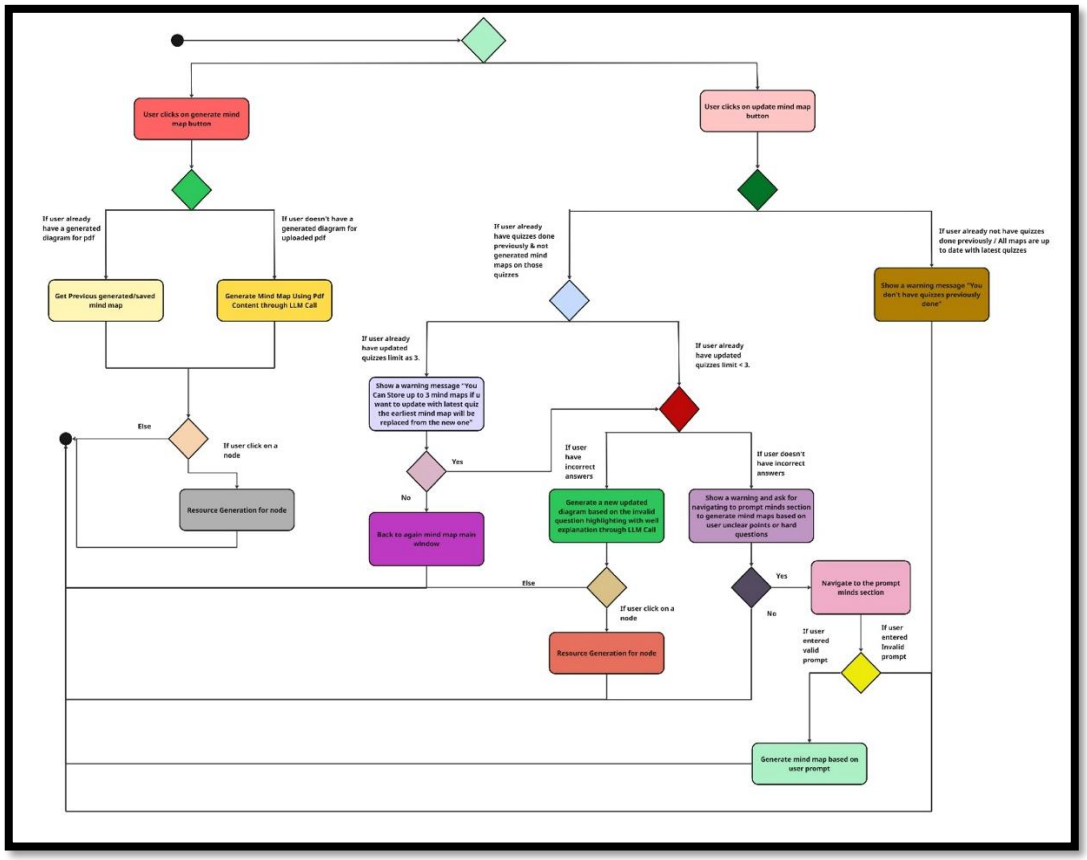


Figure 11: Adaptive Mind-Map Generation Component Implementation Sequence

3.9.1. Selection of research domains

After thorough consideration, GPT-4o-Mini as the large language model, was the best choice to be utilized.

```
const payload = {
  model: "gpt-4o-mini",
  messages: [{ role: "user", content: prompt }],
  max_tokens: 2000,
  temperature: 0.1,
};

const headers = {
  "Content-Type": "application/json",
  Authorization: `Bearer ${OPENAI_API_KEY}`,
};
```

Figure 12: Model Selection for Mind-Map

• One of the latest versions of large language models.

• Highly cost effective.

Prompt engineering practices will be used accordingly along with the GPT-4o-Mini to produce the most reliable and accurate output based on the requirement of the adaptive mind map generation.

3.9.2. Prompt engineering to generate structured version of pdf

To structure the extracted PDF content into a hierarchical JSON format, instruction-based prompting was applied as the core prompt engineering technique. A detailed and structured prompt was designed, explicitly guiding the model to group content under headings such as H1 to H6 and related paragraphs using semantic context and page numbers. The prompt also defined the exact output format to ensure database compatibility. This form of prompting allows for precise control over the model's behavior and output structure. Additionally, the integration with the Adobe PDF Extraction Tool enabled the extraction of raw content, which was then processed by the prompt to generate a meaningful and organized representation. This technique ensures consistency, accuracy, and a high level of customization when handling diverse and unstructured PDF documents.

Figure 13: Prompt – PDF Extraction

3.9.3. Prompt engineering to generate adaptive mind maps

3.9.3.1. Pdf based main mind map generation

To generate the main diagram based on the PDF content, the previously structured data is passed to the model, where the content is extracted into nodes and sub-nodes for mind map generation. (output will be further explained in Results & Discussion)

```
const MAIN_MIND_MAP_GENERATOR_PROMPT = ` You are a mind map generator. Convert the following JSON into Mermaid.js mind map input format.
For each object or array, create corresponding nodes and sub-nodes.

1. If there are long paragraphs inside the "paragraph" tags, split them into simple sentences for better readability.
2. Remove any unwanted characters like special symbols or redundant information.
3. For each paragraph, ensure that it's concise and doesn't exceed a few sentences, keeping it readable.
4. Maintain the hierarchical structure from the JSON as nodes and sub-nodes in the mind map.

This is the JSON data you need to convert to mermaid below format.:     JSON: ${JSON.stringify(extractedSections)}

Bellow I'll proivde a template how to break the things into nodes.Use this template to generate the ermaid js input format.Don't add any of your thoughts to this.
return the template only.Don't add any text or any special characters to it.Just give me the output like below format.
The format is below.Use the correct icons based on the texts.for icons only you have to use the brackets like in below format.

mindmap
root("Artificial Intelligence")
  What is AI?
    ::icon(fa fa-users)
    ("Artificial Intelligence refers to the simulation of human intelligence.")
    ("AI is programmed to think like humans and mimic their actions.")
    ("The goal of AI is to enable machines to perform tasks.")
    ("Tasks include visual perception, speech recognition, decision-making, and language translation.")
  Applications of AI
    ::icon(fa fa-play)
    ("AI is widely used in various industries.")
    ("In healthcare, AI helps diagnose diseases and predict patient outcomes.")
    ("It also helps develop personalized treatment plans.")
    ("In finance, AI is used for fraud detection.")
    ("AI also supports algorithmic trading and customer service automation.")

Do not include any brackets inside parentheses in the formatted output. This means that words or phrases within parentheses, such as (LAN), should be removed while keeping the rest of the text intact.

For example, if the original mind map node is:**
(Local Area Network (LAN) connects devices within a limited area.)

The formatted output should be:
(Local Area Network connects devices within a limited area.)

Only one root, use free FontAwesome icons, and follow node types "[", "(". No need to use "mermaid", "\`\`\`", or "graph TD". Respond only with code and syntax.`;
```

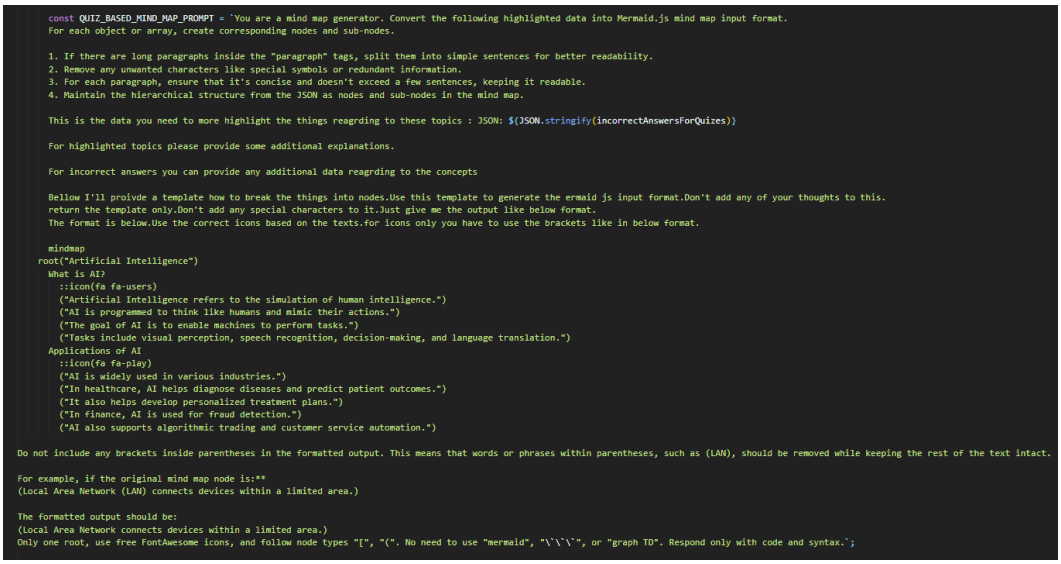
Figure 14: Prompt - Mind-Map Genaration

This prompt uses instruction-based prompting combined with few-shot prompting. A detailed instruction set guides the model's behavior, while an example output acts as a reference (few-shot) to help the model understand the expected structure and formatting of the Mermaid.js mind map.

3.9.3.2. Quiz based adaptive mind map generation

In this technique, zero-shot prompting is employed to generate mind maps from quiz-related content. The model is provided with a set of instructions without any examples of the expected input-output pairs. This enables the model to handle unseen data, such as incorrect answers from quizzes, and generate corresponding mind maps.

The instructions provided guide the model on how to process quiz data and transform it into a structured mind map format using Mermaid.js. This includes splitting long paragraphs into simple sentences for clarity, removing unwanted characters, and ensuring readability. The hierarchical structure of the data is maintained, and additional explanations are given for incorrect answers.



```
const QUIZ_BASED_MIND_MAP_PROMPT = `You are a mind map generator. Convert the following highlighted data into Mermaid.js mind map input format.
For each object or array, create corresponding nodes and sub-nodes.

1. If there are long paragraphs inside the "paragraph" tags, split them into simple sentences for better readability.
2. Remove any unwanted characters like special symbols or redundant information.
3. For each paragraph, ensure that it's concise and doesn't exceed a few sentences, keeping it readable.
4. Maintain the hierarchical structure from the JSON as nodes and sub-nodes in the mind map.

This is the data you need to more highlight the things reagrding to these topics : JSON: ${JSON.stringify(incorrectAnswersForQuizes)}

For highlighted topics please provide some additional explanations.

For incorrect answers you can provide any additional data reagrding to the concepts

Bellow I'll proivde a template how to break the things into nodes.Use this template to generate the ermaid js input format.Don't add any of your thoughts to this.
return the template only.Don't add any special characters to it.Just give me the output like below format.
The format is below.Use the correct icons based on the texts.for icons only you have to use the brackets like in below format.

 mindmap
root("Artificial Intelligence")
  What is AI?
    ::icon(fa fa-users)
    ("Artificial Intelligence refers to the simulation of human intelligence.")
    ("AI is programmed to think like humans and mimic their actions.")
    ("The goal of AI is to enable machines to perform tasks.")
    ("Tasks include visual perception, speech recognition, decision-making, and language translation.")
  Applications of AI
    ::icon(fa fa-play)
    ("AI is widely used in various industries.")
    ("In healthcare, AI helps diagnose diseases and predict patient outcomes.")
    ("It also helps develop personalized treatment plans.")
    ("In finance, AI is used for fraud detection.")
    ("AI also supports algorithmic trading and customer service automation.")

Do not include any brackets inside parentheses in the formatted output. This means that words or phrases within parentheses, such as (LAN), should be removed while keeping the rest of the text intact.

For example, if the original mind map node is:**
(Local Area Network (LAN) connects devices within a limited area.)

The formatted output should be:
(Local Area Network connects devices within a limited area.)
Only one root, use free FontAwesome icons, and follow node types "[", "(". No need to use "mermaid", "\`\`\`", or "graph TD". Respond only with code and syntax.`;
```

Figure 15: Prompt - Quiz based Mind-Map Generation

Since no specific examples are provided, the model is expected to generalize based on its prior knowledge and generate mind maps for the quiz answers provided. Figure 4.2 shows zero-shot prompting integrated with instruction-based prompting, where the model applies its learned understanding to an entirely new task, handling data it has not encountered before.

3.9.3.3. Prompt based adaptive mind map generation

3.9.3.3.1. User Prompt Validation

The QUESTION_VALIDATION prompt employs a combination of instruction-based prompting, role prompting, and chain of thought prompting to guide the model in responding to user queries based on the content of a provided PDF.

```
export const MIND_MAP_PROMPTS = {
    QUESTION_VALIDATION: `You are a study assistant helping with questions and creating mind maps.
If the user asks a question unrelated to any learning environment (e.g., general knowledge or unrelated topics), you should not create the mind maps.

You should respond if the user's question is based on the content of the provided PDF.
Study Assistant Guidelines:

Topic Relevance (PDF Content):

If the user's question relates to the content of the provided PDF, you must answer the question, even if the exact details are not explicitly mentioned in the PDF.
For example, if the PDF is about "software security aspects" and the user asks, "What are the advantages of considering software security aspects in an application?"
even if the specific advantages are not in the PDF, your answer should still be based on the overall topic of software security aspects.
Answering Based on Provided PDF:

You should answer all questions related to the content of the PDF in a detailed and comprehensive manner. If the user asks for further clarification or details,
such as asking to elaborate on specific topics like "security aspects" or requesting a breakdown of information, you should provide a thorough explanation.
For example:
If the question is, "I want to know more about security aspects in a detailed way," you should provide an in-depth explanation with ample descriptions and related information.
If the question is, "Give me a summary of software aspects," break down the summary into its core components and explain each part comprehensively.
Invalid Questions:

If the user asks a question unrelated to the content of the provided PDF, treat it as invalid. This includes questions on general knowledge, personal preferences, or unrelated topics.
For instance, questions like "What is the capital of France?" or "How do you cook spaghetti?" should be marked as invalid.
Response Handling:

If the user's question is valid (i.e., related to the content of the PDF or a related topic), respond with "Valid."
If the user's question is invalid (i.e., not related to the content of the PDF or a relevant topic), respond with "Invalid."`,
```

Figure 16: Prompt – Mind-Map Question Validation Generation

Instruction-Based Prompting provides the model with clear, structured instructions on how to handle different types of questions. The model is instructed to answer only those questions that are relevant to the content of the PDF and to give comprehensive explanations when necessary.

Role Prompting defines the model's role as a "study assistant," which helps in determining the correct behavior. The model's task is to generate answers related to the PDF's content and provide mind maps when appropriate. If the question is unrelated to the learning material, the model is instructed not to generate a mind map.

Chain of Thought Prompting helps the model process and logically reason through the steps involved in determining whether a question is valid or invalid. This ensures that

the model considers the context of the query and the PDF content before deciding on the response.

This combination of techniques ensures that the model accurately filters valid queries, responds based on the provided material, and avoids generating responses for irrelevant or off-topic questions.

3.9.3.3.2. Prompt based mind map template

The SIMPLE_MINDMAP_GENERATOR_PROMPT utilizes a combination of few-shot prompting and instruction-based prompting to guide the model in generating mind maps based on user input.

```
SIMPLE_MINDMAP_GENERATOR_PROMPT :
`Create a mermaid mindmap based on user input like these examples:
brainstorming mindmap
mindmap
\t\troot(("leisure activities weekend"))
\t\t\t["spend time with friends"]
\t\t\t\t::icon(fafa fa-users)
\t\t\t\t\t("action activities")
\t\t\t\t\t\t::icon(fafa fa-play)
\t\t\t\t\t\t\t\t("dancing at night club")
\t\t\t\t\t\t\t\t("going to a restaurant")
\t\t\t\t\t\t\t\t("go to the theater")
\t\t\t\t["spend time your self"]
\t\t\t\t\t::icon(fa fa-fa-user)
\t\t\t\t\t\t("meditation")
\t\t\t\t\t\t::icon(fa fa-om)
\t\t\t\t\t\t\t("\`take a sunbath 🟠\`")
\t\t\t\t\t\t\t("reading a book")
\t\t\t\t\t\t\t::icon(fa fa-book)
text summary mindmap:
Barack Obama (born August 4, 1961) is an American politician who served as the 44th president of the United States from 2009 to 2017. A member of the Democratic Party, he was the first African-American president of the United States.
mindmap
\troot("Barack Obama")
\t\t("Born August 4, 1961")
\t\t::icon(fa fa-baby-carriage)
\t\t("American Politician")
\t\t\t::icon(fa fa-flag)
\t\t\t\t("44th President of the United States")
\t\t\t\t\t("2009 - 2017")
\t\t("Democratic Party")
\t\t\t::icon(fa fa-democrat)
\t\t("First African-American President")
cause and effects mindmap:
mindmap
\troot("Landlord sells apartment")
\t\t::icon(fa fa-sell)
\t\t("Renter must be notified of sale")
\t\t::icon(fa fa-envelope)
\t\t\t("Tenants may feel some uncertainty")
\t\t\t::icon(fa fa-question-circle)
\t\t("Notice periods must be observed")
\t\t::icon(fa fa-calendar)
\t\t\t("Landlord can submit notice of termination for personal use")
\t\t\t::icon(fa fa-home)
\t\t\t\t("Tenant has to look for a new apartment")
\t\t\t\t::icon(fa fa-search)
\t\t("New owner")
\t\t::icon(fa fa-user)
\t\t\t("New owner takes over existing rental agreement")
\t\t\t::icon(fa fa-file-contract)
\t\t\t\t("Tenant keeps previous apartment")
\t\t\t\t::icon(fa fa-handshake)
\t\t\t("New owner terminates newly concluded lease")
\t\t\t::icon(fa fa-ban)
\t\t\t\t("Tenant has to look for a new apartment")
\t\t\t\t::icon(fa fa-search)
Only one root, use free FontAwesome icons, and follow node types "[", "(". No need to use "mermaid", "\`\`\`", or "graph TD". Respond only with code and syntax.`
};
```

Figure 17: Prompt – Simple Mind-Map Generation

Few-Shot Prompting is used here by providing the model with multiple example formats for generating mind maps. These examples, such as "brainstorming mindmap," "text summary mindmap," and "cause and effects mindmap," serve as guides to help the model understand the structure and output style it should follow

when creating new mind maps from user input. This gives the model a sense of the desired output format, even for new or unseen data.

Instruction-Based Prompting provides clear, direct guidelines for how the model should process the input. The prompt explicitly tells the model to follow a specific structure (e.g., using nodes and sub-nodes, employing Font Awesome icons, and adhering to a specific node format). The model is instructed to output mind maps in a format with one root and follow the provided examples without additional comments or symbols.

This combination of techniques ensures that the model is able to generate mind maps in a consistent, structured way based on various types of input, including brainstorming ideas, text summaries, and cause-effect relationships.

3.9.3.4. Resource handling

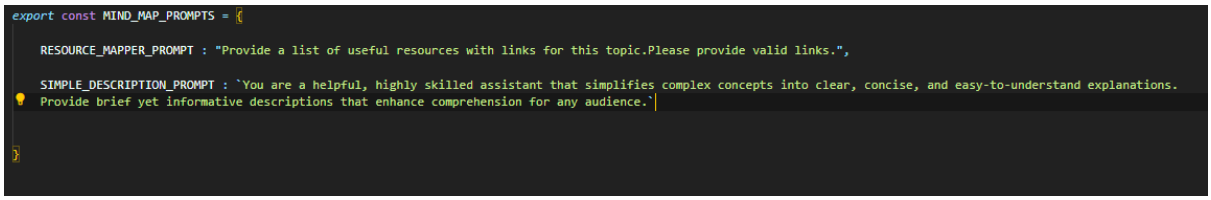


Figure 18: Prompt – Resource handling

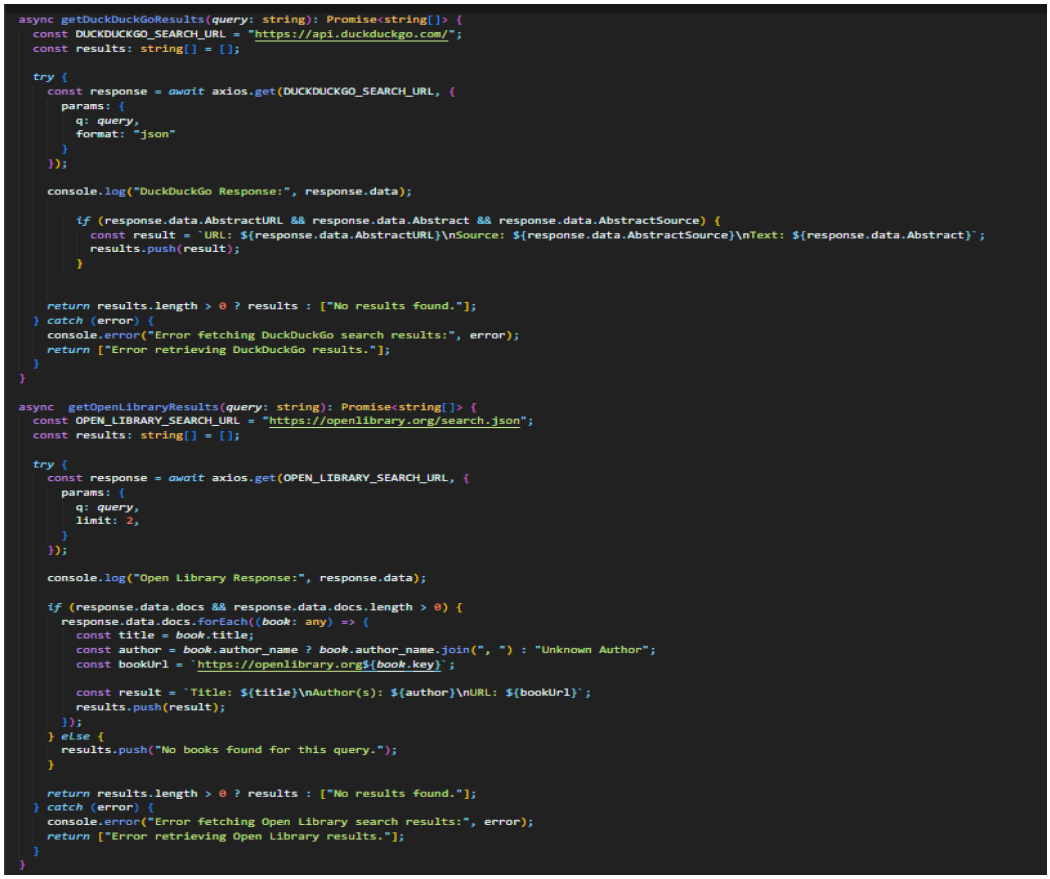
The RESOURCE_MAPPER_PROMPT and SIMPLE_DESCRIPTION_PROMPT use zero-shot prompting and instruction-based prompting. In these prompts, the model is not provided with specific examples but is instead given clear instructions on how to perform the task. For the RESOURCE_MAPPER_PROMPT, the model is simply asked to list useful resources with valid links, while the SIMPLE_DESCRIPTION_PROMPT guides the model to simplify complex concepts into clear, concise explanations. These approaches allow the model to handle tasks based on the instructions without needing prior examples.

3.9.3.4.1. API Integration for Resource Linking.

The resource linking process retrieves relevant resources based on the user's query. It first searches through different search engines like Google Search API, DuckDuckGo, and Open Library to find links related to the topic. If no relevant results are found, the system falls back on using OpenAI's API to generate resource suggestions related to the query. This ensures that users receive a variety of resources both from online databases and AI-generated suggestions tailored to the topic they are interested in.

```
private async getNodeResources(nodeText: string): Promise<string[]> {
  const results: string[] = [];

  // Array of search engines
  const searchEngines = [
      { url: this.googleSearchUrl, params: { key: GOOGLE_SEARCH_API_KEY, cx: YOUTUBE_SEARCH_ENGINE_ID, q: `"${nodeText}"`, num: 5 } },
      { url: this.googleSearchUrl, params: { key: GOOGLE_SEARCH_API_KEY, cx: GOOGLE_SEARCH_ENGINE_ID, q: `"${nodeText}"`, num: 5 } },
  ];

  try {
      for (const searchEngine of searchEngines) {
          const response = await axios.get(searchEngine.url, { params: searchEngine.params });
          const items = response.data.items;

          if (items && items.length > 0) {
              items.forEach((item: any) => {
                  // Basic filtering based on domain
                  if (!results.includes(item.link) && this.isRelevantDomain(item.link)) {
                      results.push(item.link);
                  }
                  results.push(item.link)
              });
          }
      }

      if (results.length === 0) {
        console.log("No results found. Fetching from OpenAI...");
        const openAiResponse = await this.getAiGeneratedResources(nodeText);
        return openAiResponse.length > 0 ? openAiResponse : ["No resources available."];
    }

      // Limit to 5 results overall
      return results.slice(0, 5);
  } catch (error) {
      console.error("Error fetching resources from Google Custom Search API:", error);
      return ["Error retrieving resources."];
  }
}
```

Figure 19: Google Search API Integration



```
async getDuckDuckGoResults(query: string): Promise<string[]> {
  const DUCKDUCKGO_SEARCH_URL = "https://api.duckduckgo.com/";
  const results: string[] = [];

  try {
    const response = await axios.get(DUCKDUCKGO_SEARCH_URL, {
      params: {
        q: query,
        format: "json"
      }
    });

    console.log("DuckDuckGo Response:", response.data);

        if (response.data.AbstractURL && response.data.Abstract && response.data.AbstractSource) {
          const result = `URL: ${response.data.AbstractURL}\nSource: ${response.data.AbstractSource}\nText: ${response.data.Abstract}`;
          results.push(result);
        }


    return results.length > 0 ? results : ["No results found."];
  } catch (error) {
    console.error("Error fetching DuckDuckGo search results:", error);
    return ["Error retrieving DuckDuckGo results."];
  }
}

async  getOpenLibraryResults(query: string): Promise<string[]> {
  const OPEN_LIBRARY_SEARCH_URL = "https://openlibrary.org/search.json";
  const results: string[] = [];

  try {
    const response = await axios.get(OPEN_LIBRARY_SEARCH_URL, {
      params: {
        q: query,
        limit: 2,
      }
    });

    console.log("Open Library Response:", response.data);

    if (response.data.docs && response.data.docs.length > 0) {
      response.data.docs.forEach((book: any) => {
        const title = book.title;
        const author = book.author_name ? book.author_name.join(", ") : "Unknown Author";
        const bookUrl = `https://openlibrary.org${book.key}`;

        const result = `Title: ${title}\nAuthor(s): ${author}\nURL: ${bookUrl}`;
        results.push(result);
      });
    } else {
      results.push("No books found for this query.");
    }

    return results.length > 0 ? results : ["No results found."];
  } catch (error) {
    console.error("Error fetching Open Library search results:", error);
    return ["Error retrieving Open Library results."];
  }
}
```

Figure 20: LLM Integration for Resource Generation

3.9.4. Web Interfaces for mind maps

Figure 21: UI - Mind-Map

Figure 22: UI - Quiz Based Mind Map

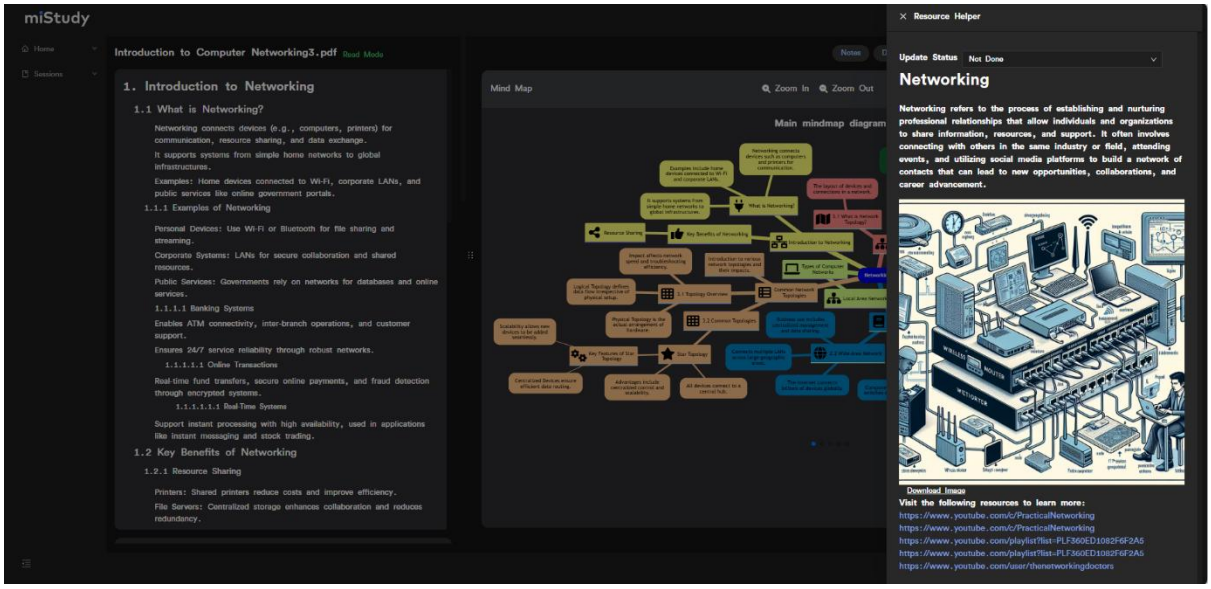


Figure 23: Mind-Map Resourse Linking

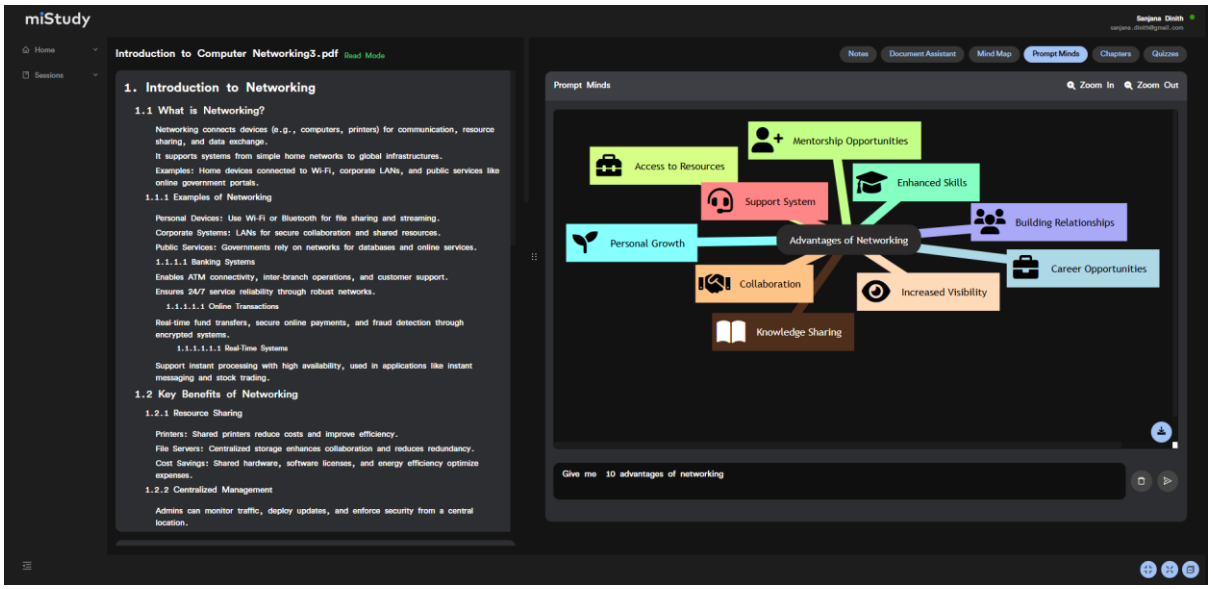


Figure 24: UI - Prompt Mind-Maps

3.9.5. Web Interfaces for note cards and chapter view

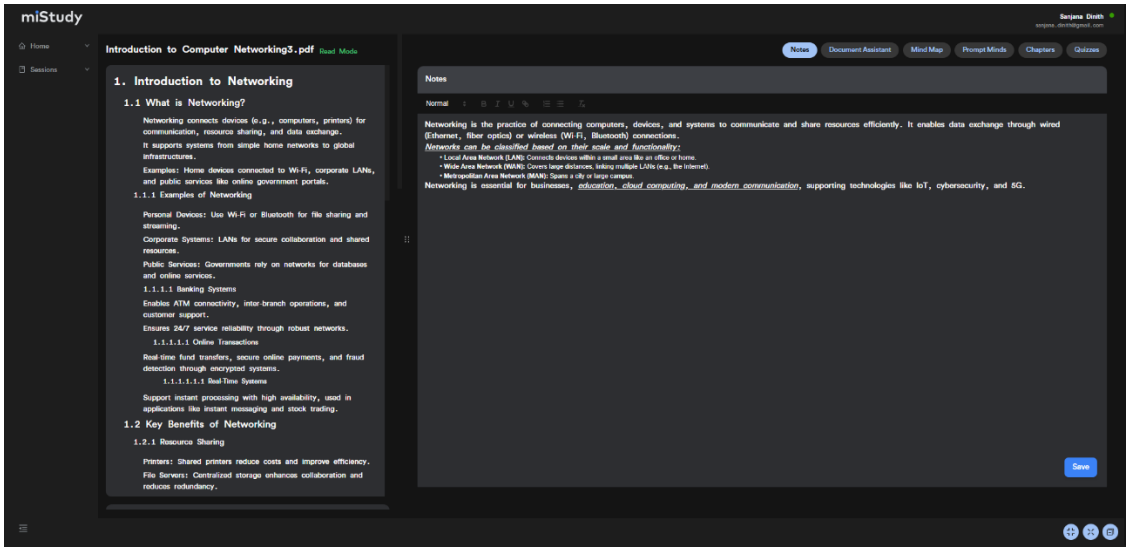


Figure 25: UI - Note Cards

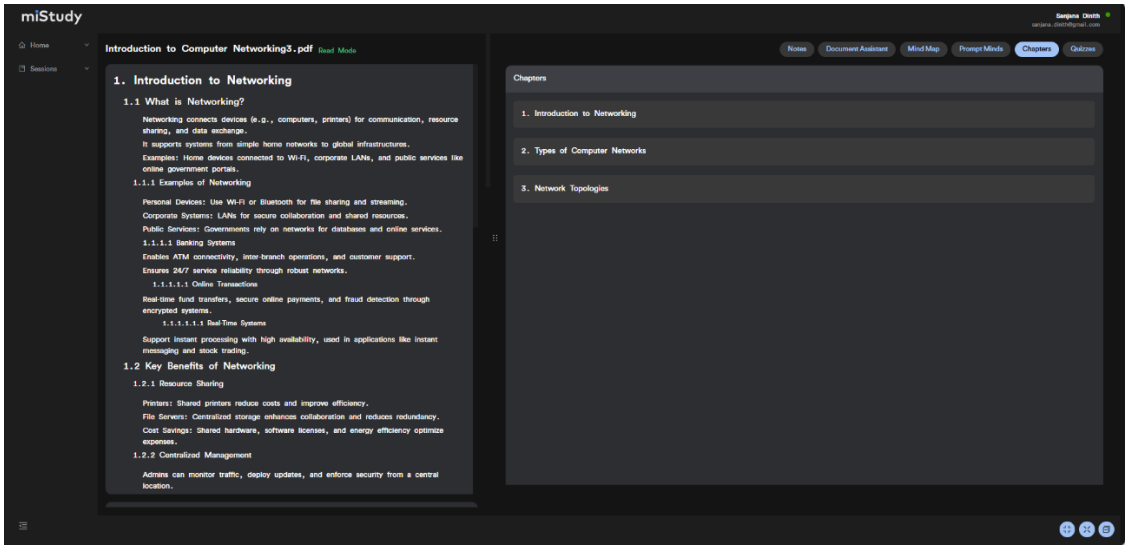


Figure 26: UI -Chapter view

3.10. Testing

To ensure the reliability and effectiveness of the AI-driven Mind Map Generation System, a comprehensive testing phase was conducted. The testing process involved evaluating the system's mind map generation accuracy, content structuring efficiency, user prompt validation, and resource enrichment module performance. Various testing techniques, including functional testing, performance testing, and UAT, were employed to validate the system's ability to generate accurate, structured, and personalized mind maps based on user input and quiz interactions.

The primary focus of testing was to assess the system's ability to correctly extract structured data from PDF documents, process user inputs, and generate accurate visual representations using AI-powered techniques. Additionally, the integration of external resource enrichment using Google APIs, Bing search, and DALL·E was tested to verify the correctness and relevance of the retrieved data. The testing process also included quiz-based performance tracking, ensuring that the system accurately updates mind maps based on students' strengths and weaknesses.

3.10.1. Test Plan

The test plan outlines the structured methodology used for evaluating the LMS with AI-driven Mind Map Generation. The testing process was designed to cover all critical functionalities, including user interactions, AI-based data processing, prompt validation, and dynamic content enrichment. The plan involved defining test objectives, setting up test conditions, preparing test data, and executing various test cases to ensure that the system meets all functional and performance requirements.

The test plan also ensured that the system functions smoothly under different user inputs, whether structured PDF content, custom prompts, or quiz-based updates. Multiple test cases were executed to verify that the mind maps evolve correctly over time, accurately reflecting user progress.

3.10.2. Test Strategy

The test strategy focused on ensuring system stability, accuracy, and efficiency. The approach involved unit testing, integration testing, and system testing, along with error logging and defect tracking. Continuous testing iterations were conducted in collaboration with developers, testers, and domain experts to enhance the system's scalability and adaptability.

Key testing techniques included:

- Functional Testing: Verifying the mind map generation process, from data extraction to AI-driven visualization.
- Performance Testing: Measuring the response time of AI processing, ensuring that large datasets and external resource integration do not slow down the system.
- UAT: Evaluating the system's usability and effectiveness based on real student interactions and learning experiences.
- Error Handling and Edge Case Testing: Ensuring that incorrect or ambiguous user inputs do not generate flawed mind maps or crash the system.

3.10.3. Test Case Design

Table 2: Verify the register functionality of the system

Test Case ID	01
Test Case	Verify the register functionality of the system
Test Case Scenario	Users register to the system
Input	Full Name –Sanaya Samadhi Phone Number - 0743099876 Email – sanayasamadhi@gmail.com

	Password – Test@123 Confirm Password – Test@123
Expected Output	If registration is successful, the user should navigate to the login screen
Actual Result	After submitting the user details, the user is navigated to the login screen.
Status (Pass / Fail)	Pass

Table 3: Verify the PDF content extraction and structuring process

Test Case ID	02
Test Case	Verify the PDF content extraction and structuring process.
Test Case Scenario	A user uploads a study material PDF file for mind map generation.
Input	PDF File: "Machine Learning Basics.pdf"
Expected Output	System extracts and structures key topics into a hierarchical format.
Actual Result	The extracted content is correctly structured into sections
Status (Pass / Fail)	Pass

Table 4: Verify the AI-driven mind map generation from structured content.

Test Case ID	03
Test Case	Verify the AI-driven mind map generation from structured content.
Test Case Scenario	System generates a mind map based on structured data extracted from PDF.
Input	Structured data extracted from PDF

Expected Output	The system generates an accurate mind map with clear relationships.
Actual Result	Mind map successfully generated with correct topic hierarchy.
Status (Pass / Fail)	Pass

Table 5: Verify user prompt-based mind map generation.

Test Case ID	04
Test Case	Verify user prompt-based mind map generation.
Test Case Scenario	A user provides a custom topic prompt for mind map creation according to the pdf content
Input	Prompt: "Explain Deep Learning with examples."
Expected Output	The system generates a structured mind map covering key aspects.
Actual Result	Mind map correctly generated with relevant subtopics and examples.
Status (Pass / Fail)	Pass

Table 6: Verify user prompt-based mind map generation for invalid contexts

Test Case ID	05
Test Case	Verify user prompt-based mind map generation for invalid contexts
Test Case Scenario	A user provides a custom topic prompt for mind map creation not according to the pdf content
Input	Prompt: "Explain What are the best places in Sri Lanka for vacationing?"

Expected Output	The system replies with a message please come up with a valid question
Actual Result	Mind maps cannot be generated and users can see a message “This is not regarding to the pdf content. Come up with valid question”
Status (Pass / Fail)	Pass

Table 7: Verify quiz-based mind map updates.

Test Case ID	06
Test Case	Verify quiz-based mind map updates.
Test Case Scenario	A user completes a quiz, and the system updates the mind map based on incorrect answers.
Input	Quiz Results: 3 incorrect answers in "Neural Networks"
Expected Output	The system generates the mind map by emphasizing weak areas.
Actual Result	Mind map successfully generated, highlighting weak concepts.
Status (Pass / Fail)	Pass

Table 8: Verify integration of external resource enrichment.

Test Case ID	07
Test Case	Verify integration of external resource enrichment.
Test Case Scenario	The system fetches additional learning materials from external sources.
Input	Node content “What is networking”
Expected Output	The system retrieves relevant valid articles, videos, and AI-generated

	diagrams and description about key concepts.
Actual Result	External valid resources successfully integrated and displayed.
Status (Pass / Fail)	Pass

Table 9: Verify the replacement of the earliest mind map

Test Case ID	08
Test Case	Verify the replacement of the earliest mind map when exceeding the storage limit.
Test Case Scenario	The system allows storing up to three mind maps. When a user completes a new quiz, the earliest stored mind map is replaced with the latest one.
Input	1.User has three stored mind maps. 2.The user completes a new quiz. 3. The system attempts to store the newly updated mind map.
Expected Output	The earliest stored mind map is removed, and the newly updated mind map is stored. Warning should be displayed to the user.
Actual Result	The system successfully removed the oldest mind map and added the new one and a warning is displayed as expected.
Status (Pass / Fail)	Pass

3.11. Commercialization

The Adaptive Mind Map Generation module within the miStudy LMS is designed to revolutionize how learners interact with and understand educational content. By leveraging advanced AI techniques such as RAG, LLMs, and performance data analysis, this component offers dynamic, personalized, and visually structured learning paths. It helps address the growing need for individualized study experiences and effective content summarization in higher education.

This feature holds significant potential to be commercialized as a premium offering within both institutional LMS platforms and standalone SaaS learning tools, capitalizing on the global demand for AI-driven educational personalization.

3.11.1. Market Research and Analysis

- **Target Audience**:
    - Universities and educational institutions seeking intelligent tools to enhance learner engagement.
    - EdTech companies, digital course providers, and LMS vendors looking to integrate adaptive learning solutions.
- **Market Needs Addressed**:
    - The lack of personalization in conventional LMS platforms.
    - The challenge of visualizing complex academic material for improved retention.

- The demand for AI-powered tools to support hybrid and online learning environments.

3.11.2. Product Development

- The Adaptive Mind Map Generator is a web-based microservice that integrates seamlessly with the miStudy LMS.
- Core features include:
    - PDF content extraction and topic summarization.
    - AI-generated mind maps tailored to individual learning performance.
    - Quiz-based feedback loops that dynamically alter the mind map structure.
    - Prompt-based mind map generation for concept reinforcement and curiosity-driven exploration.
- The system supports storing up to four adaptive mind maps per user, intelligently replacing outdated maps with new insights based on quiz performance.

3.11.3. Monetization Strategy

- **Premium Institutional Offering**:
    - Bundle adaptive mind map generation as a premium module within the LMS offering.
    - Subscription-based pricing tiers for academic institutions based on the number of enrolled users.
- **Freemium to Premium for Individuals**
    - Free access with limited features (e.g., PDF-based mind maps).
    - Paid subscription unlocks quiz-adaptive mind maps, prompt-based mind maps, resource-enriched nodes, and performance analytics.
- **Custom Licensing**

- Offer exclusive licensing to EdTech firms or LMS vendors for white-labeled integration.

3.11.4. Marketing and Promotion

- **B2B Institutional Promotion**:
  - Showcase at global EdTech events, conferences, and university IT summits.
  - Collaborate with academic influencers and LMS product evaluators.
- **B2C Awareness Campaigns**:
  - Leverage YouTube videos, webinars, and content marketing on adaptive learning strategies.
  - Promote via LinkedIn and educational platforms targeting teachers, tutors, and higher-ed students.
- **Use Cases & Testimonials**:
  - Develop case studies showing improved academic performance through personalized mind mapping.
  - Share student feedback and visual transformation of learning journeys.

3.11.5. Continuous Innovation and Iteration

- Ongoing refinement of mind map accuracy through user feedback and performance metrics.
- Interactive node editing and seamless reordering capabilities enable users to tailor and organize their mind maps for an optimized learning experience.

4. RESULTS AND DISCUSSION

4.1. Results

4.1.1. Results of Large Language Model Evaluation

To enhance personalized adaptive mind map generation, various models and techniques were explored and evaluated. The implementation involved integrating RAG, LLMs, and external knowledge retrieval techniques to create adaptive, user-centered mind maps. The performance was assessed based on accuracy, adaptability, and efficiency in generating contextually relevant learning maps.

Table 4.1 shows the details of the evaluated LLM models and the benchmark scores which the respective models have produced, along with the cost of the usage of the model.

Table 10: details of the evaluated LLM models and the benchmark scores

Model	**GPT-3.5 Turbo**	**GPT-4**	**GPT-4 Turbo**	**GPT-4o Mini**	**LLaMA 3.1 8B Instruct**	**Gemini Flash**
Provider	OpenAI	OpenAI	OpenAI	OpenAI	Meta	Google
Input Context Window	4,096 tokens	8K / 32K tokens (variant)	128K tokens	128K tokens	128K tokens	1M tokens
	4,096 tokens					

Max Output Tokens		4,096 tokens	4,096 tokens	16.4K tokens	2,048 tokens	8,192 tokens
Release Date	Nov 28, 2022	Mar 14, 2023	Apr 9, 2024	Jul 18, 2024	Jul 23, 2024	May 14, 2024
Knowledge Cutoff	Sep 2021	Sep 2023	Dec 2023	Oct 2023	Dec 2023	Nov 2023
Open Source	No	No	No	No	Yes	No
Input Cost	$0.50	$3.00	$10.00	$0.15	$0.09	$0.13
Output Cost	$1.50	$6.00	$30.00	$0.60	$0.09	$0.38
MMLU Benchmark	70.0	86.4	86.7	82.0	66.7	78.9

HumanEval Benchmark	60.3	85.0	88.2	87.2	34.8	74.3
HellaSwag Benchmark	85.5	94.5	95.3	87.2	80.01	86.5

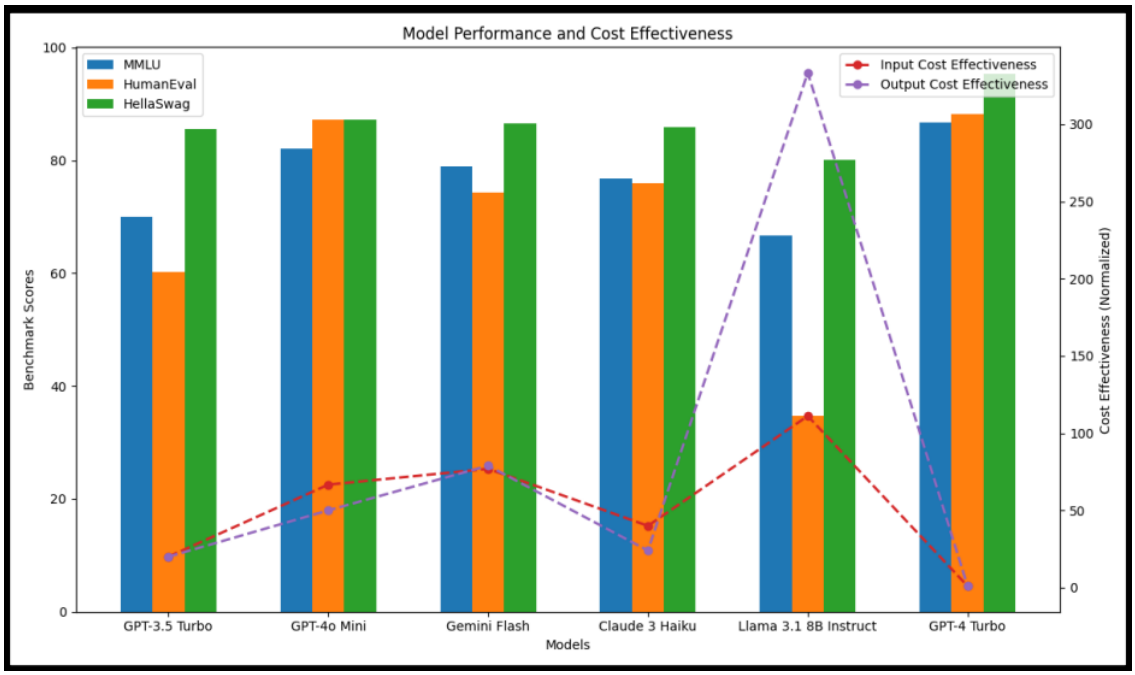


Figure 27 Model Performance

This derived bar chart shows that the Llama 3.1 8B Instruct LLM is very cost effective, yet the performance is drastically low.

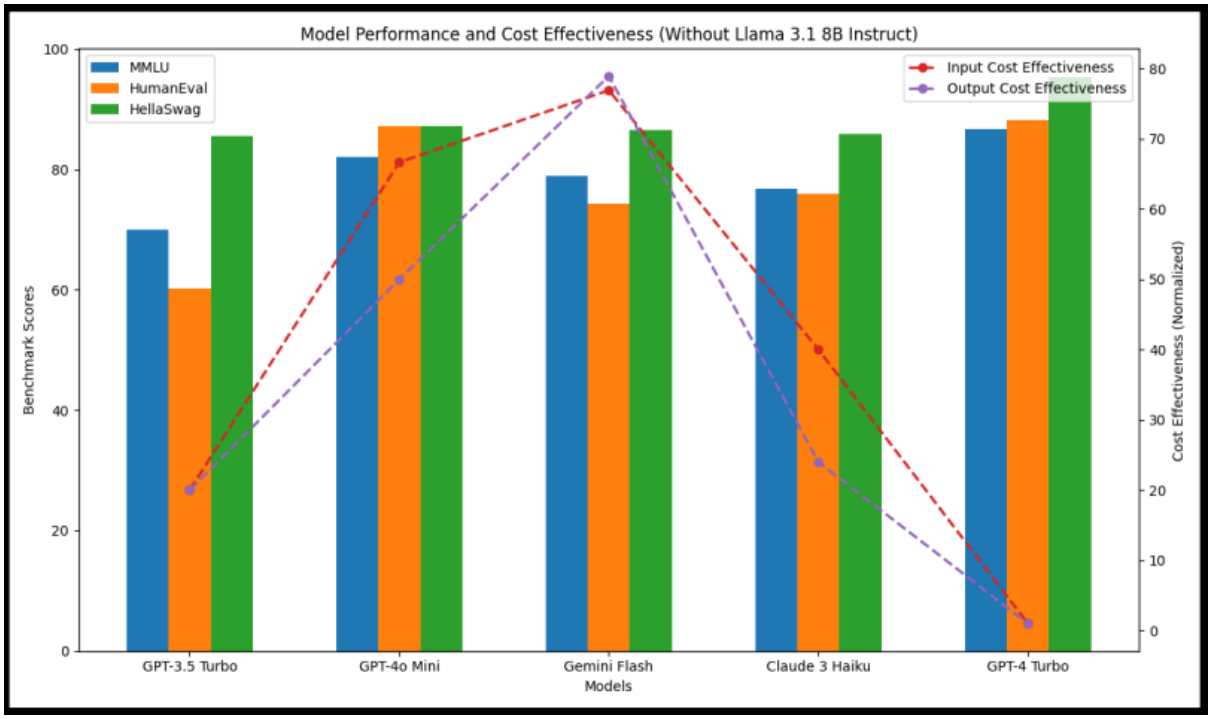


Figure 28 Model performance and cost effectiveness

The derived bar graph visualizes the cost-effectiveness and performance of various large language models (LLMs). While GPT-4 Turbo demonstrates the highest benchmark performance across MMLU, HumanEval, and HellaSwag, it is also the most expensive model, offering very poor cost-effectiveness. On the other hand, Gemini Flash is the most cost-effective model, especially for input and output processing, but it slightly underperforms in benchmark scores compared to other top-tier models.

Models such as GPT-3.5 Turbo and Claude 3 Haiku offer a balance between cost and performance but still fall short either in accuracy or flexibility. GPT-4o Mini stands out as the most balanced model, providing high benchmark performance close to GPT-4 Turbo while maintaining substantially lower costs, making it ideal for real-time and large-scale tasks like mind map generation. For this reason, GPT-4o Mini is the most practical and efficient choice among the evaluated models.

4.1.2. Results of Prompt Engineering for Mind Map Generation

Prompt engineering played a vital role in shaping the responses of the AI model to align with the functional requirements of the adaptive mind map generator. Different prompting styles and templates were explored for various scenarios ranging from content-driven mind map creation to quiz-adaptive visualizations and question-based contextual mapping. The table below summarizes the techniques evaluated and applied.

Table 11: Promting Techniques

Prompting Techniques	**Application Scenario**
Instruction-Based Prompting	PDF-based mind map generation, Quiz-based adaptive mind maps, Prompt-based mind map creation
Few-shot Prompting	PDF-based mind map generation ,Quiz-based adaptive mind maps
Zero-shot Prompting	Resource-enhanced mind map construction and description generations
Role Prompting	Resource-enhanced mind map construction and description generations
Chain of Thought Prompting	Question validation prompts

4.1.2.1. Few-shot Prompting and Instruction-based Prompting for pdf-based mind map generation

Few-shot prompting, along with instruction-based prompting, was utilized for converting structured PDF content into a hierarchical mind map in the Mermaid.js format. The prompt provided the following detailed instructions:

- Break down long paragraphs into shorter, readable sentences
- Maintain semantic clarity and structural integrity
- Remove special characters and nested brackets
- Ensure only a single root node with relevant Font Awesome icons

The results obtained using the main prompt, `MAIN_MIND_MAP_GENERATOR_PROMPT`, showed high consistency in generating clear and visually structured mind maps with high semantic alignment. Below is the structure of the primary prompt used:

const **MAIN_MIND_MAP_GENERATOR_PROMPT** = ` You are a mind map generator. Convert the following JSON into Mermaid.js mind map input format.

For each object or array, create corresponding nodes and sub-nodes.

1. If there are long paragraphs inside the "paragraph" tags, split them into simple sentences for better readability.

2. Remove any unwanted characters like special symbols or redundant information.

3. For each paragraph, ensure that it's concise and doesn't exceed a few sentences, keeping it readable.

4. Maintain the hierarchical structure from the JSON as nodes and sub-nodes in the mind map.

This is the JSON data you need to convert to mermaid below format.: JSON: ${JSON.stringify(extractedSections)}

Bellow I'll provide a template how to break the things into nodes. Use this template to generate the mermaid js input format. Don't add any of your thoughts to this.

return the template only. Don't add any text or any special characters to it. Just give me the output like below format.

The format is below. Use the correct icons based on the texts. For icons only, you have to use the brackets like in below format.

mindmap

root("Artificial Intelligence")

What is AI?

::icon(fa fa-users)

("Artificial Intelligence refers to the simulation of human intelligence.")

("AI is programmed to think like humans and mimic their actions.")

("The goal of AI is to enable machines to perform tasks.")

("Tasks include visual perception, speech recognition, decision-making, and language translation.")

Applications of AI

::icon(fa fa-play)

("AI is widely used in various industries.")

("In healthcare, AI helps diagnose diseases and predict patient outcomes.")

("It also helps develop personalized treatment plans.")

("In finance, AI is used for fraud detection.")

("AI also supports algorithmic trading and customer service automation.")

Do not include any brackets inside parentheses in the formatted output. This means that words or phrases within parentheses, such as (LAN), should be removed while keeping the rest of the text intact.

For example, if the original mind map node is:

(Local Area Network (LAN) connects devices within a limited area.)

The formatted output should be:

(Local Area Network connects devices within a limited area.)

Only one root, use free Font Awesome icons, and follow node types "[", "(". No need to use "mermaid", "\`\`\`", or "graph TD". Respond only with code and syntax. `;

```
mindmap
  root(Networking)
    Introduction to Networking
      ::icon(fa fa-network-wired)
      What is Networking?
        (Networking connects devices e.g., computers, printers for communication.)
        (It supports systems from simple home networks to global infrastructures.)
        (Examples include home devices connected to Wi-Fi and corporate LANs.)
      Key Benefits of Networking
    Types of Computer Networks
      ::icon(fa fa-sitemap)
      Local Area Network
    Centralized Management
      ::icon(fa fa-cogs)
      (Admins can monitor traffic, deploy updates, and enforce security.)
    History of Networking
      ::icon(fa fa-history)
      Telegraph and Telephone
        (Telegraph enabled coded message transmission Morse Code.)
        (Telephone allowed real-time voice communication over long distances.)
      Early Innovations
        (Morse Code & Switching Stations streamlined long-distance communication.)
      Long-Distance Communication
        (Telegraph Lines & Submarine Cables connected cities and continents.)
      Packet Switching
        (Introduced in the 1960s with ARPANET.)
        (Improved efficiency and reliability, forming the foundation of modern internet.)
    Network Concepts
      ::icon(fa fa-th)
      Introduction to LAN
        (Definition connects devices within a limited area like a home or office.)
        (Components include switches, routers, and devices.)
      Shared Resources
        (Shared Drives enable secure access and collaboration.)
        (Benefits include real-time updates and enhanced team productivity.)
      Wide Area Network
        (Definition connects multiple LANs across large geographic areas.)
        (Internet connects billions of devices globally.)
      WAN Business Use
        (Multinational Operations enable centralized management.)
        (Data Sharing allows real-time synchronization and cloud integration.)
    Network Topologies
      ::icon(fa fa-sitemap)
      What is Network Topology?
        (Definition is the layout of devices and connections in a network.)
        (Impact affects network speed and troubleshooting efficiency.)
      Common Topologies
        (Star Topology has centralized control.)
        Advantages
          (Centralized control simplifies management.)
          (Failure of individual devices doesn't disrupt the network.)
          (Highly scalable for growing networks.)
        Key Features of Star Topology
          (Centralized Devices ensure efficient data routing.)
          (Scalability allows new devices to be added seamlessly.)
```

Figure 29: PDF Based mind map generation output format

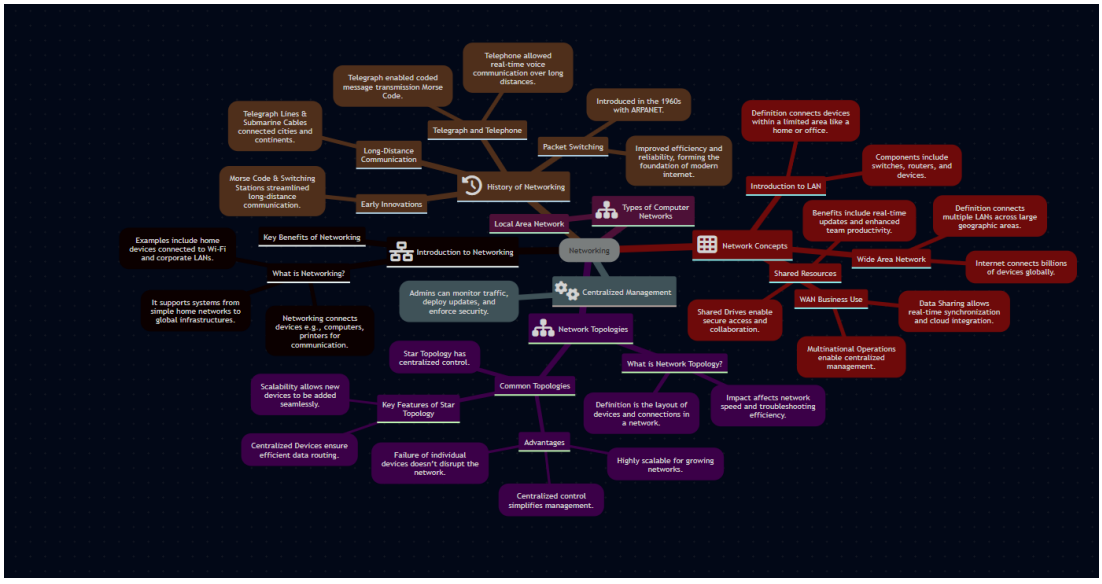


Figure 30: PDF Based mind map genaration output

4.1.2.2. Prompt Engineering Techniques Identified in Quiz based Mind Map Generation

Several prompt engineering techniques were tested in the context of quiz-based adaptive mind map generation. These techniques include:

1. **Instruction-based Prompting**

    - The prompt clearly instructs the AI to parse quiz-related data and generate a mind map using a predefined structure. It ensures clarity and consistency in the output.

2. **Structured Output Prompting**

    - This technique guides the AI to output mind maps in a strict structure, ensuring no additional text or characters are included in the result. The model follows specific rules on node types and icon usage.

3. **Zero-shot Prompting**

    - Zero-shot prompting allows the AI to handle new and unseen scenarios. In this case, the model was asked to generate mind maps based on incorrect quiz answers and explanations, without having seen specific examples beforehand. This approach tested the AI's ability to generalize.

const **QUIZ_BASED_MIND_MAP_PROMPT**= `You are a mind map generator. Convert the following highlighted data into Mermaid.js mind map input format.

For each object or array, create corresponding nodes and sub-nodes.

1. If there are long paragraphs inside the "paragraph" tags, split them into simple sentences for better readability.

2. Remove any unwanted characters like special symbols or redundant information.

3. For each paragraph, ensure that it's concise and doesn't exceed a few sentences, keeping it readable.

4. Maintain the hierarchical structure from the JSON as nodes and sub-nodes in the mind map.

This is the data you need to more highlight the things regarding to these topics: JSON: ${JSON.stringify(incorrectAnswersForQuizes)}

For highlighted topics please provide some additional explanations.

For incorrect answers you can provide any additional explanations regarding to the concepts

Bellow I'll provide a template how to break the things into nodes.Use this template to generate the mermaid js input format. Don’t add any of your thoughts to this.Return the template only. Don’t add any special characters to it. Just give me the output like below format.

The format is below. Use the correct icons based on the texts. For icons only you have to use the brackets like in below format.

mindmap

root("Artificial Intelligence")

What is AI?

::icon(fa fa-users)

("Artificial Intelligence refers to the simulation of human intelligence.")

("AI is programmed to think like humans and mimic their actions.")

("The goal of AI is to enable machines to perform tasks.")

("Tasks include visual perception, speech recognition, decision-making, and language translation.")

Applications of AI

::icon(fa fa-play)

("AI is widely used in various industries.")

("In healthcare, AI helps diagnose diseases and predict patient outcomes.")

("It also helps develop personalized treatment plans.")

("In finance, AI is used for fraud detection.")

("AI also supports algorithmic trading and customer service automation.")

Do not include any brackets inside parentheses in the formatted output. This means that words or phrases within parentheses, such as (LAN), should be removed while keeping the rest of the text intact.

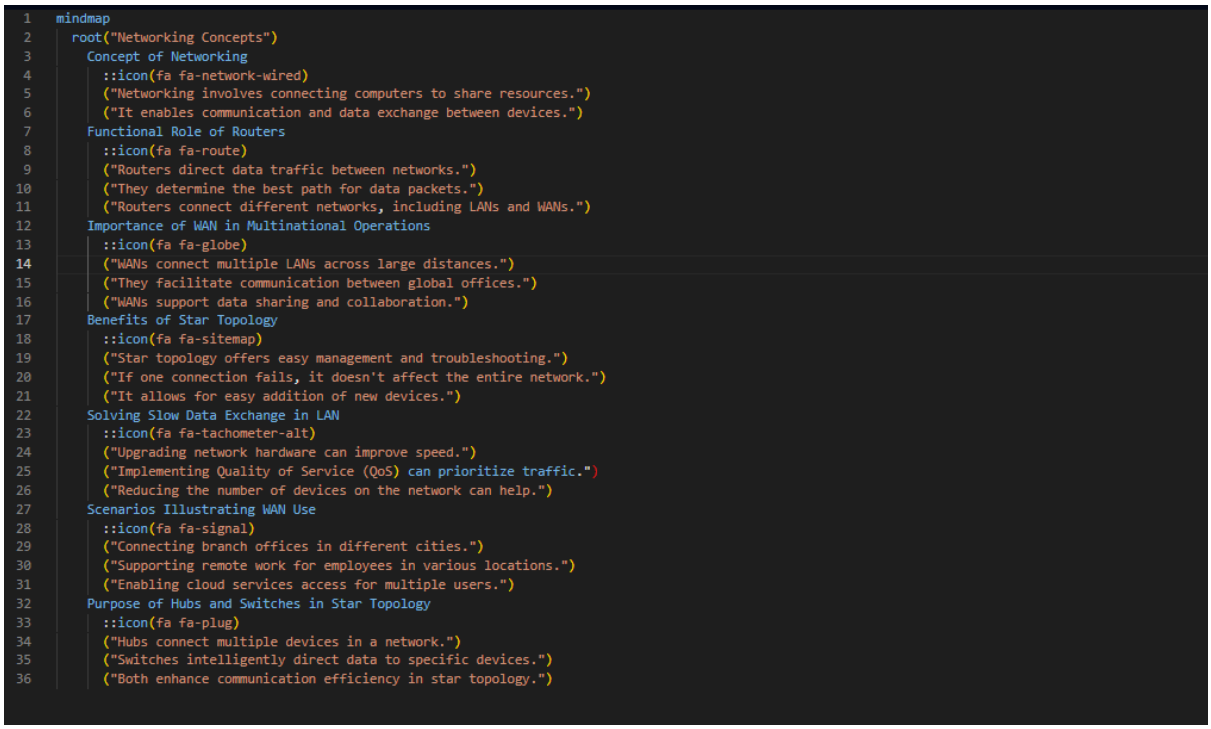
For example, if the original mind map node is:

(Local Area Network (LAN) connects devices within a limited area.)

The formatted output should be:

(Local Area Network connects devices within a limited area.)

Only one root, use free Font Awesome icons, and follow node types "[", "(". No need to use "mermaid", "\`\`\`", or "graph TD". Respond only with code and syntax. `;



```
mindmap
  root("Networking Concepts")
    Concept of Networking
      ::icon(fa fa-network-wired)
      ("Networking involves connecting computers to share resources.")
      ("It enables communication and data exchange between devices.")
    Functional Role of Routers
      ::icon(fa fa-route)
      ("Routers direct data traffic between networks.")
      ("They determine the best path for data packets.")
      ("Routers connect different networks, including LANs and WANs.")
    Importance of WAN in Multinational Operations
      ::icon(fa fa-globe)
      ("WANs connect multiple LANs across large distances.")
      ("They facilitate communication between global offices.")
      ("WANs support data sharing and collaboration.")
    Benefits of Star Topology
      ::icon(fa fa-sitemap)
      ("Star topology offers easy management and troubleshooting.")
      ("If one connection fails, it doesn't affect the entire network.")
      ("It allows for easy addition of new devices.")
    Solving Slow Data Exchange in LAN
      ::icon(fa fa-tachometer-alt)
      ("Upgrading network hardware can improve speed.")
      ("Implementing Quality of Service (QoS) can prioritize traffic.")
      ("Reducing the number of devices on the network can help.")
    Scenarios Illustrating WAN Use
      ::icon(fa fa-signal)
      ("Connecting branch offices in different cities.")
      ("Supporting remote work for employees in various locations.")
      ("Enabling cloud services access for multiple users.")
    Purpose of Hubs and Switches in Star Topology
      ::icon(fa fa-plug)
      ("Hubs connect multiple devices in a network.")
      ("Switches intelligently direct data to specific devices.")
      ("Both enhance communication efficiency in star topology.")
```

Figure 31: Quiz Based mind map generation output format

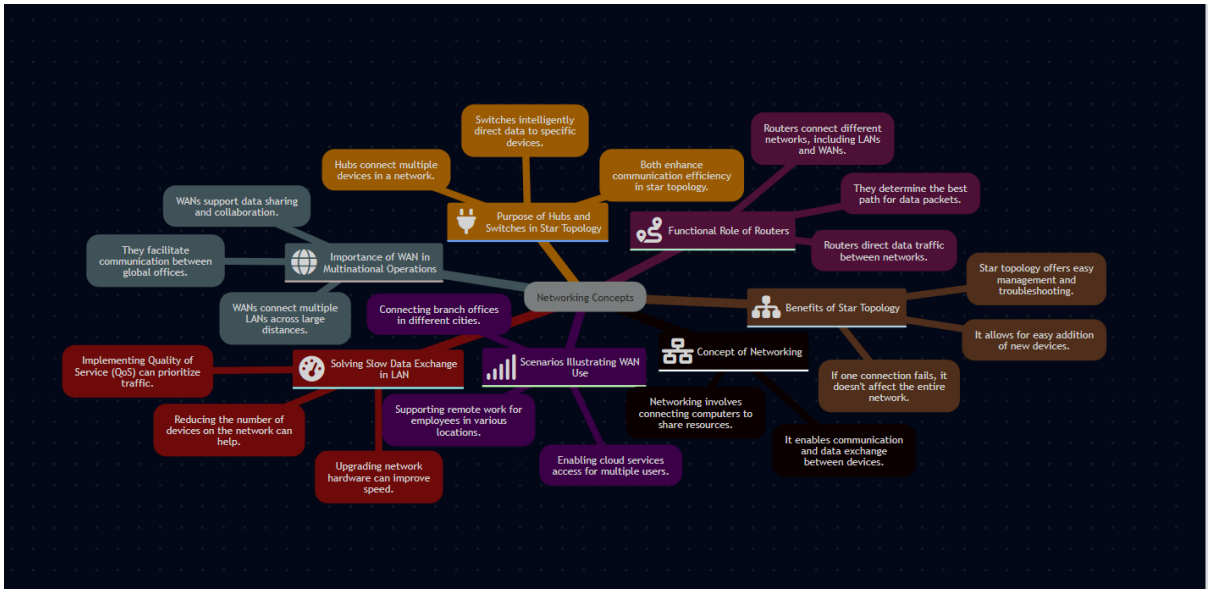


Figure 32: Quiz Based mind map generation output

4.1.2.3. Instruction-based Prompting and Role-Prompting & Chain of thought Prompting for question validation

In this technique, the model is given clear instructions defining its role, which helps in determining valid and invalid responses. For example, in a quiz-based learning scenario, the model's role as a "study assistant" guides it to generate mind maps based on the user's input. With Chain of Thought (CoT) prompting, the model first analyzes the user's question, checking if it relates to the content of the provided PDF. If the question is relevant, the model answers in detail; if it's unrelated, it responds with "Invalid." This approach ensures accurate and context-aware responses while maintaining focus on the intended learning material.

QUESTION_VALIDATION: `You are a study assistant helping with questions and creating mind maps based on those questions.

If the user asks a question unrelated to any learning environment (e.g., general knowledge or unrelated topics), you should not create the mind maps.

You should respond if the user's question is based on the content of the provided PDF.

Study Assistant Guidelines:

Topic Relevance (PDF Content):

If the user's question relates to the content of the provided PDF, you must answer the question, even if the exact details are not explicitly mentioned in the PDF. For example, if the PDF is about "software security aspects" and the user asks, "What are the advantages of considering software security aspects in an application?" even if the specific advantages are not in the PDF, your answer should still be based on the overall topic of software security aspects.

Answering Based on Provided PDF:

You should answer all questions related to the content of the PDF in a detailed and comprehensive manner. If the user asks for further clarification or details, such as asking to elaborate on specific topics like "security aspects" or requesting a breakdown of information, you should provide a thorough explanation.

For example:

If the question is, "I want to know more about security aspects in a detailed way," you should provide an in-depth explanation with ample descriptions and related information.

If the question is, "Give me a summary of software aspects," break down the summary into its core components and explain each part comprehensively.

Invalid Questions:

If the user asks a question unrelated to the content of the provided PDF, treat it as invalid. This includes questions on general knowledge, personal preferences, or unrelated topics.

For instance, questions like "What is the capital of France?" or "How do you cook spaghetti?" should be marked as invalid.

Response Handling:

If the user's question is valid (i.e., related to the content of the PDF or a related topic), respond with "Valid."

If the user's question is invalid (i.e., not related to the content of the PDF or a relevant topic), respond with "Invalid."`,

4.1.2.4. Few-shot Prompting and Instruction-based Prompting for simple mind map generation

This is the main technique used in this prompt.

It gives multiple concrete examples of expected input-output behavior:

- Brainstorming mind map
- Text summary mind map
- Cause and effects mind map

These examples show structure, formatting, icon usage, and node hierarchy.

Specific instructions are included, such as:

- *"Only one root"*
- *"Use free Font Awesome icons"*
- *"Follow node types '[', '('"*
- *"No need to use 'mermaid', '```', or 'graph TD'"*

SIMPLE_MINDMAP_GENERATOR_PROMPT :

`Create a mermaid mindmap based on user input like these examples:

brainstorming mindmap

mindmap

\t\troot(("leisure activities weekend"))

\t\t\t\t["spend time with friends"]

\t\t\t\t::icon(fafa fa-users)

\t\t\t\t\t\t("action activities")

\t\t\t\t\t\t::icon(fafa fa-play)

\t\t\t\t\t\t\t\t("dancing at night club")

\t\t\t\t\t\t\t\t("going to a restaurant")

\t\t\t\t\t\t\t\t("go to the theater")

\t\t\t\t["spend time your self"]

\t\t\t\t::icon(fa fa-fa-user)

\t\t\t\t\t\t("meditation")

\t\t\t\t\t\t::icon(fa fa-om)

\t\t\t\t\t\t("\`take a sunbath ⛱\`")

\t\t\t\t\t\t("reading a book")

\t\t\t\t\t::icon(fa fa-book)

text summary mindmap:

Barack Obama (born August 4, 1961) is an American politician who served as the 44th president of the United States from 2009 to 2017. A member of the Democratic Party, he was the first African-American president of the United States.

mindmap

\troot("Barack Obama")

\t\t("Born August 4, 1961")

\t\t::icon(fa fa-baby-carriage)

\t\t("American Politician")

\t\t\t::icon(fa fa-flag)

\t\t\t\t("44th President of the United States")

\t\t\t\t\t("2009 - 2017")

\t\t("Democratic Party")

\t\t\t::icon(fa fa-democrat)

\t\t("First African-American President")

cause and effects mindmap:

mindmap

\troot("Landlord sells apartment")

```
\t\t::icon(fa fa-sell)

\t\t("Renter must be notified of sale")

\t\t::icon(fa fa-envelope)

\t\t\t("Tenants may feel some uncertainty")

\t\t\t::icon(fa fa-question-circle)

\t\t("Notice periods must be observed")

\t\t::icon(fa fa-calendar)

\t\t\t("Landlord can submit notice of termination for personal use")

\t\t\t::icon(fa fa-home)

\t\t\t\t("Tenant has to look for a new apartment")

\t\t\t\t::icon(fa fa-search)

\t\t("New owner")

\t\t::icon(fa fa-user)

\t\t\t\t("New owner takes over existing rental agreement")

\t\t\t\t::icon(fa fa-file-contract)

\t\t\t\t\t\t("Tenant keeps previous apartment")

\t\t\t\t\t\t::icon(fa fa-handshake)

\t\t\t\t("New owner terminates newly concluded lease")
```

\t\t\t\t::icon(fa fa-ban)

\t\t\t\t\t\t("Tenant has to look for a new apartment")

\t\t\t\t\t\t::icon(fa fa-search)

Only one root, use free Font Awesome icons, and follow node types "[", "(". No need to use "mermaid", "\`\`\`", or "graph TD". Respond only with code and syntax.
`};

```
mindmap
   root("Advantages of Networking")
      ["Career Opportunities"]
      ::icon(fa fa-briefcase)
      ["Access to Resources"]
      ::icon(fa fa-toolbox)
      ["Knowledge Sharing"]
      ::icon(fa fa-book-open)
      ["Building Relationships"]
      ::icon(fa fa-users)
      ["Increased Visibility"]
      ::icon(fa fa-eye)
      ["Support System"]
      ::icon(fa fa-headset)
      ["Collaboration"]
      ::icon(fa fa-handshake)
      ["Mentorship Opportunities"]
      ::icon(fa fa-user-plus)
      ["Enhanced Skills"]
      ::icon(fa fa-graduation-cap)
      ["Personal Growth"]
      ::icon(fa fa-seedling)
```

Figure 33: Prompt Based Mind Map genatation output format

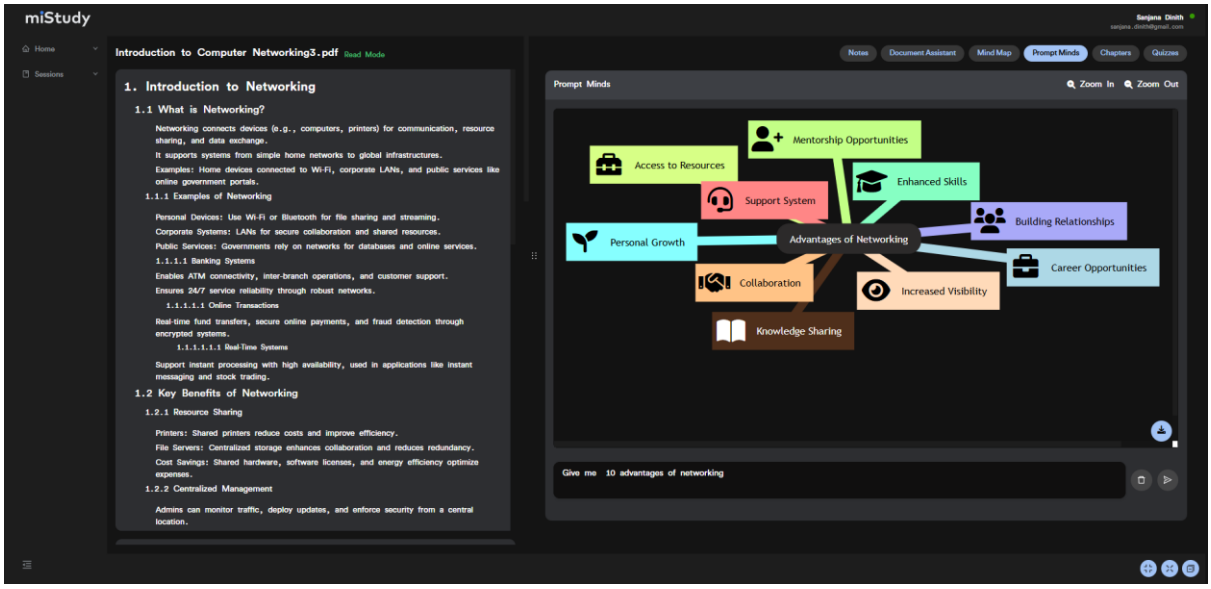


Figure 34: : Prompt Based Mind Map generation output format

4.1.2.5. Zero-shot Prompting and Instruction based Prompting for resource generations

Zero-shot prompting allows the AI to handle tasks it has not been explicitly trained on, such as listing useful resources for a specific topic. Instruction-based prompts guide the AI on how to behave and approach the task, even without prior examples.

RESOURCE_MAPPER_PROMPT: "Provide a list of useful resources with links for this topic. Please provide valid links.", **SIMPLE_DESCRIPTION_PROMPT**: "You are a helpful, highly skilled assistant that simplifies complex concepts into clear, concise, and easy-to-understand explanations. Provide brief yet informative descriptions that enhance comprehension for any audience."

Description output, Node Text: Networking

Networking refers to the process of establishing and nurturing professional relationships that allow individuals and organizations to share information, resources, and support. It often involves connecting with others in the same industry or field, attending events, and utilizing social media platforms to build a network of contacts that can lead to new opportunities, collaborations, and career advancement.

4.2. Research Findings

The research findings have revealed a critical gap in personalized learning resources and the difficulty many learners face in organizing and understanding vast academic content. Despite the availability of digital learning platforms, students often struggle to comprehend core concepts due to the absence of adaptive tools that align with their individual learning progress. The integration of adaptive mind map generation into Learning Management Systems (LMS) effectively addresses this gap by transforming static educational content into interactive, personalized visual structures that evolve based on the learner's performance and behavior.

Through the use of LLMs, quiz analysis, and resource linking, the system dynamically generates mind maps that highlight weak points and reinforce understanding with relevant learning materials. This not only enhances engagement but also ensures that students receive targeted guidance in areas they are struggling with. By simplifying complex topics into manageable visual nodes, learners are more likely to retain information, make meaningful connections, and improve their academic outcomes.

Moreover, the adaptive nature of the system means that mind maps update in real-time with each quiz attempt or prompt input, promoting continuous learning and immediate feedback. These findings underscore the effectiveness of intelligent visual tools in modern education and highlight the potential for such systems to support both self-directed learning and institutional education. With continued enhancement and

collaboration with educators, adaptive mind maps can redefine digital learning by making it more responsive, efficient, and learner centered.

4.3. Discussions

This research aimed to design and develop a web-based Learning Management System (LMS) with a core focus on adaptive mind map generation to enhance personalized learning experiences. The system was specifically developed to bridge the gap in conventional e-learning platforms, which often lack adaptability and visual learning tools that cater to individual learner needs. Through in-depth background research and stakeholder engagement, essential system requirements were identified ranging from personalized learning pathways to interactive visual content delivery which informed the overall system design and architecture.

The research involved implementing and evaluating a range of technologies including Large Language Models (LLMs), prompt engineering techniques, Retrieval-Augmented Generation (RAG), Library Integrations, and quiz-based learning analytics. These technologies were strategically integrated to enable the system to generate customized mind maps based on user input, uploaded study material (PDFs), and quiz performance. The system was designed to update mind maps in real-time, ensuring that learners receive focused content based on their evolving learning patterns and weak points.

A key insight of the study was the realization that static study materials are insufficient for students with diverse learning needs. Adaptive mind map generation, powered by AI, helped simplify complex topics by breaking them into visual nodes that reflected the most relevant subtopics for each user. This not only enhanced learner engagement but also improved content retention. The integration of prompt-based and quiz-based mind maps proved highly effective in identifying knowledge gaps and providing dynamic revision paths.

Furthermore, the system’s storage logic where up to four mind maps are maintained (one from the original content and three based on quiz results) ensured continuity in

learning while maintaining performance. Users could access previous mind maps for reference or automatically update them based on the latest assessments, contributing to a truly adaptive learning environment.

The system's architecture demonstrated how AI-powered educational tools can shift traditional LMSs from static repositories to interactive, learner-centric platforms. With the help of visualization libraries and performance monitoring, the application also delivered real-time feedback loops and resource suggestions, supporting students throughout their academic journey.

In conclusion, this study highlights the feasibility and effectiveness of adaptive mind map generation in modern educational systems. The findings confirm that such systems can significantly enhance learning outcomes by promoting personalized, interactive, and visually structured learning. Future work may involve integrating feedback from educators and deploying mobile accessibility to further improve reach and inclusivity.

5. CONCLUSION

To conclude, this study highlights the effectiveness of integrating AI-driven adaptive mind maps in enhancing personalized learning experiences. By leveraging advanced technologies such as RAG and LLMs, the system provides dynamic, personalized learning materials that adapt based on user performance. The use of AI to generate adaptive mind maps ensures that the learning content is tailored to individual needs, improving student engagement and knowledge retention. Furthermore, the integration of quiz results into the system allows for real-time adjustments to the mind maps, addressing areas of weakness and promoting a more effective learning process.

The implementation of features such as customizable mind maps and resource linking further supports the flexibility and scalability of the system, making it a powerful tool for diverse learning environments. The results demonstrate the potential of such adaptive systems in transforming traditional educational practices, making learning more efficient and personalized.

Future developments will focus on expanding the system's capabilities to support a broader range of learning materials, enhancing the accuracy of mind map generation, and further optimizing performance. Collaboration with educators and continuous refinement of the AI models will be essential for increasing the system's effectiveness and ensuring its widespread adoption in educational settings. Ultimately, this adaptive learning system holds the potential to significantly improve the quality of education, offering scalable and personalized learning experiences for students worldwide.

REFERENCES

[1] S. Dolnicar and B. Grün, "Does one size fit all? The suitability of answer formats for different constructs measured," *Australasian Marketing Journal*, vol. 17, no. 1, pp. 58–64, 2009.

[2] T. Buzan, *Mind Map Mastery: The Complete Guide to Learning and Using the Most Powerful Thinking Tool in the Universe*, Mumbai, India: Jaico Publishing House, 2024.

[3] M. A. Razafinirina, W. G. Dimbisoa, and T. Mahatody, "Pedagogical alignment of large language models (LLM) for personalized learning: A survey, trends and challenges," *Journal of Intelligent Learning Systems and Applications*, vol. 16, no. 4, pp. 448–480, 2024.

[4] T. Abbas and A. Abbas, "AI-powered concept mapping in collaborative and individual learning: Enhancing ICT-based reading comprehension," 2025.

[5] L. Hu et al., "Text-graph enhanced knowledge graph representation learning," *Frontiers in Artificial Intelligence*, vol. 4, Art. no. 697856, 2021.

[6] R. Kudelić, M. Maleković, and A. Lovrenčić, "Mind map generator software," in *Proc. 2012 IEEE Int. Conf. Computer Science and Automation Engineering (CSAE)*, vol. 3, pp. 123–127, 2012.

[7] S. S. Bae, O.-H. Kwon, S. Chandrasegaran, and K.-L. Ma, "Spinneret: Aiding creative ideation through non-obvious concept associations," in *Proc. 2020 CHI Conf. Human Factors in Computing Systems*, pp. 1–13, 2020.

[8] M. Elhoseiny and A. Elgammal, "Text to multi-level mindmaps: A novel method for hierarchical visual abstraction of natural language text," *Multimedia Tools and Applications*, vol. 75, pp. 4217–4244, 2016.

[9] Y. Shi, H. Yang, Y. Dou, and Y. Zeng, "Effects of mind mapping-based instruction on student cognitive learning outcomes: A meta-analysis," *Asia Pacific Education Review*, vol. 24, no. 3, pp. 303–317, 2023.

[10] A. Okada and T. Connolly, "Designing open educational resources through knowledge maps to enhance meaningful learning," *International Journal of Learning Technology*, vol. 15, no. 7, pp. 209–220, 2008.

[11] R. Kudelić, M. Maleković, and A. Lovrenčić, "Mind map generator software," in *Proc. 2012 IEEE Int. Conf. Computer Science and Automation Engineering (CSAE)*, 2012.

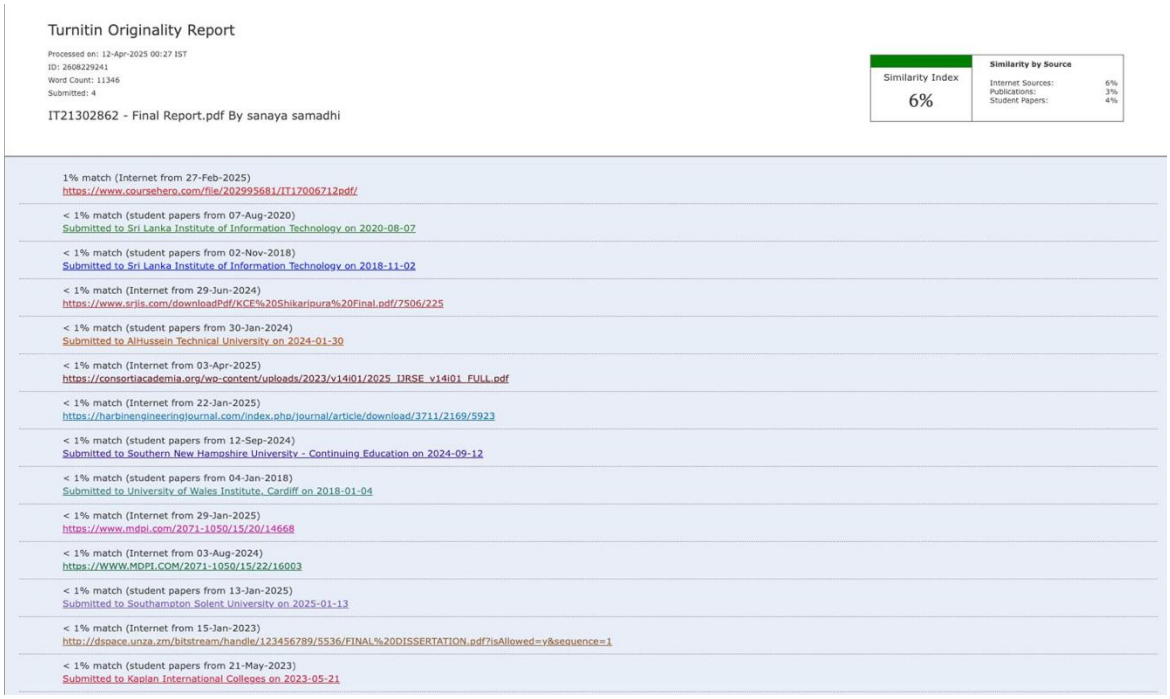
[12] M. Elhoseiny and A. Elgammal, "Text to multi-level mindmaps: A novel method for hierarchical visual abstraction of natural language text," *Multimedia Tools and Applications*, vol. 75, pp. 4217–4244, 2016.

[13] Y. Shi, H. Yang, Y. Dou, and Y. Zeng, "Effects of mind mapping-based instruction on student cognitive learning outcomes: A meta-analysis," *Asia Pacific Education Review*, vol. 24, no. 3, pp. 303–317, 2023.

[14] A. Okada and T. Connolly, "Designing open educational resources through knowledge maps to enhance meaningful learning," *International Journal of Learning Technology*, vol. 15, no. 7, pp. 209–220, 2008.

APPENDICES

Plagiarism Report



Turnitin Originality Report

Processed on: 12-Apr-2025 00:27 IST
ID: 2608229241
Word Count: 11346
Submitted: 4

IT21302862 - Final Report.pdf By sanaya samadhi

Similarity Index	Similarity by Source	
6%	Internet Sources:	6%
	Publications:	3%
	Student Papers:	4%

1% match (Internet from 27-Feb-2025)
https://www.coursehero.com/file/202995681/IT17006712pdf/

< 1% match (student papers from 07-Aug-2020)
Submitted to Sri Lanka Institute of Information Technology on 2020-08-07

< 1% match (student papers from 02-Nov-2018)
Submitted to Sri Lanka Institute of Information Technology on 2018-11-02

< 1% match (Internet from 29-Jun-2024)
https://www.srjis.com/downloadPdf/KCE%20Shikaripura%20Final.pdf/7506/225

< 1% match (student papers from 30-Jan-2024)
Submitted to AlHussein Technical University on 2024-01-30

< 1% match (Internet from 03-Apr-2025)
https://consortiacademia.org/wp-content/uploads/2023/v14i01/2025_IJRSE_v14i01_FULL.pdf

< 1% match (Internet from 22-Jan-2025)
https://harbinengineeringjournal.com/index.php/journal/article/download/3711/2169/5923

< 1% match (student papers from 12-Sep-2024)
Submitted to Southern New Hampshire University - Continuing Education on 2024-09-12

< 1% match (student papers from 04-Jan-2018)
Submitted to University of Wales Institute, Cardiff on 2018-01-04

< 1% match (Internet from 29-Jan-2025)
https://www.mdpi.com/2071-1050/15/20/14668

< 1% match (Internet from 03-Aug-2024)
https://WWW.MDPI.COM/2071-1050/15/22/16003

< 1% match (student papers from 13-Jan-2025)
Submitted to Southampton Solent University on 2025-01-13

< 1% match (Internet from 15-Jan-2023)
http://dspace.unza.zm/bitstream/handle/123456789/5536/FINAL%20DISSERTATION.pdf?isAllowed=y&sequence=1

< 1% match (student papers from 21-May-2023)
Submitted to Kaplan International Colleges on 2023-05-21