A Synopsis of Project on

LUMIS : LLM Based Unified Multimodal Intelligent System

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Engineering

in

Computer Science and Engineering (Data Science)

by

Dalbirsingh Matharu(21107005)
Umesh Pawar(21107014)
Shreyas Patil(21107061)
Vedant Parulekar(21107034)

Under the Guidance of

Prof. Sarala Mary
Prof. Richa Singh

Department of Computer Science and Engineering (Data Science)
A.P. Shah Institute of Technology
G.B.Road,Kasarvadavli, Thane(W)-400615
UNIVERSITY OF MUMBAI

Academic Year 2024-2025

Approval Sheet

This Project Synopsis Report entitled ***"LUMIS : LLM Based Unified Multimodal Intelligent System"*** Submitted by ***"Dalbirsingh Matharu"(21107005), "Umesh Pawar"(21107014), "Shreyas Patil"(21107061), "Vedant Parulekar"(21107034)***is approved for the partial fulfillment of the requirenment for the award of the degree of ***Bachelor of Engineering*** in ***Compurt Science and Engineering (Data Science)*** from ***University of Mumbai***.

(Prof. Richa Singh)
Co-Guide

(Prof. Sarala Mary)
Guide

Prof. Anagha Aher
HOD, Data Science

Place:A.P.Shah Institute of Technology, Thane
Date:

CERTIFICATE

This is to certify that the project entitled ***"LUMIS : LLM Based Unified Multimodal Intillegent System"*** submitted by ***"Dalbirsingh Matharu" (21107005), "Umesh Pawar" (21107014), "Shreyas Patil" (21107061), "Vedant Parulkear" (21107034)*** for the partial fulfillment of the requirement for award of a degree ***Bachelor of Engineering*** in ***Computer Science and Engineering (Data Science)***,to the University of Mumbai,is a bonafide work carried out during academic year 2024-2025.

(Prof. Richa Singh)	(Prof. Sarala Mary)
Co-Guide	Guide

Prof. Anagha Aher	Dr. Uttam D.Kolekar
HOD, Data Science	Principal

External Examiner(s)	Internal Examiner(s)
1.	1.
2.	2.

Place:A.P.Shah Institute of Technology, Thane
Date:

Acknowledgement

We have great pleasure in presenting the synopsis report on **LUMIS : LLM Based Unified Multimodal Intelligent System** We take this opportunity to express our sincere thanks towards our guide **Prof.Sarala Mary** & Co-Guide **Prof.Richa Singh** for providing the technical guidelines and suggestions regarding line of work. We would like to express our gratitude towards his constant encouragement, support and guidance through the development of project.

We thank **Prof. Anagha Aher** Head of Department for her encouragement during the progress meeting and for providing guidelines to write this report.

We express our gratitude towards BE project co-ordinator **Prof. Poonam Pangarkar, Prof.Ashwini Rahude**, for being encouraging throughout the course and for their guidance.

We also thank the entire staff of APSIT for their invaluable help rendered during the course of this work. We wish to express our deep gratitude towards all our colleagues of APSIT for their encouragement.

Dalbirsingh Matharu
(21107005)

Umesh Pawar
(21107014)

Shreyas Patil
(21107061)

Vedant Parulekar
(21107034)

Declaration

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, We have adequately cited and referenced the original sources. We also declare that We have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Dalbirsingh Matharu(21107005)

Umesh Pawar(21107014)

Shreyas Patil(21107061)

Vedant Parulekar(21107034)

Date:

Abstract

Project Lumis is an innovative initiative focused on developing a unified multimodal intelligent system powered by advanced large language models (LLMs). It aims to integrate a wide range of capabilities, including text generation from various inputs and images, text summarization, YouTube video transcription, website transcription, image-to-text conversion, and multiformat file retrieval through APIs such as OpenAI and Gemini.
At the core of LUMIS is a suite of intelligent AI agents designed with the LangChain framework. This allows the system to function as an adaptive assistant that learns from user interactions, makes informed decisions, and automates complex tasks. Emphasizing user-friendliness and voice assistance, Lumis provides a versatile solution for diverse multimodal tasks.
Key features include an intelligent code assistant that offers real-time code comprehension, error detection, and suggestion generation, seamlessly integrating with popular code editors to enhance developer productivity. Additionally, LUMIS features SQL integration for streamlined database interactions and an Air Canva functionality for creating visually appealing content effortlessly. Continuous feature updates will ensure that LUMIS remains at the forefront of technological advancements, effectively meeting the evolving needs of its users.

Keywords: LLM, multimodal, text generation, text summarization, video transcription, website transcription, image-to-text, file aggregation, AI agents, LangChain, Gemini, OpenAI, intelligent assistant, automation, adaptive learning, user-friendly, voice assistant, code assistant, SQL integration, Air Canva.

Contents

List of Figures

List of Tables

List of Abbreviations

LUMIS:	LLM based Unified Multimodal Integrated System
RAG:	Retrieval Augmented Generation
TAPAS:	Technical Assistance Platform for Advanced Solution
AI:	Artificial Intelligence

Chapter 1

Introduction

LUMIS is an innovative initiative aimed at transforming the landscape of software development through the application of advanced large language models (LLMs). As software development grows increasingly complex, developers face a myriad of challenges, including the integration of various tools, the need for quick error resolution, and the management of large volumes of information. To address these challenges, there is a pressing need for a comprehensive solution that integrates multiple functionalities into a cohesive platform. **LUMIS** responds to this need by enhancing developer productivity and streamlining workflows, ultimately facilitating a more efficient development process. Central to **LUMIS** is the

integration of diverse functionalities specifically designed to meet the evolving needs of developers and content creators alike. The system features robust text generation capabilities, enabling users to produce high-quality written content with ease. Additionally, **LUMIS** incorporates transcription services that convert audio and video materials into editable text formats, ensuring that users can efficiently manage and utilize their information. The platform also offers image-to-text conversion, allowing users to extract and manipulate data from visual content effectively. By leveraging APIs from industry leaders such as OpenAI and Gemini, **LUMIS** ensures that users benefit from state-of-the-art technology, significantly improving the overall effectiveness and versatility of the system. One of the standout features

of **LUMIS** is its real-time code assistant, which provides immediate feedback on coding errors as they occur. This capability not only reduces the time developers spend debugging but also allows them to maintain focus on their core tasks, thereby enhancing productivity. Furthermore, the integration of SQL functionalities streamlines database management by automating query generation and extraction, enabling developers to efficiently handle large datasets without unnecessary complexity. This feature significantly empowers users, allowing them to execute data-related tasks with confidence and precision.

Committed to continuous improvement, **LUMIS** evolves alongside user needs through regular updates that introduce new features and enhancements. This proactive approach ensures that the platform remains relevant and effective in a rapidly changing technological landscape. Moreover, **LUMIS** is designed with a user-friendly interface that promotes accessibility for developers of all skill levels, fostering an intuitive experience that minimizes the learning curve often associated with advanced tools.

In summary, **LUMIS** stands out as a comprehensive platform that integrates advanced functionalities—text generation, transcription, image-to-text conversion, real-time code assistance, and SQL management—to address the multifaceted challenges of modern application development. By focusing on usability and prioritizing continuous updates, **LUMIS** enhances productivity and streamlines workflows, making it an essential resource for developers seeking to navigate the complexities of today's technology landscape.

1.1 Motivation

Project Lumis is motivated by the clear absence of a fully integrated large language model (LLM) solution that provides a comprehensive suite of functionalities for developers and content creators. In today's fast-paced software development landscape, developers often rely on multiple, disjointed tools for various tasks, including code assistance, SQL management, text generation, and content creation. This fragmentation results in inefficiencies, disrupted workflows, and increased complexity, as developers must switch between different systems to complete their projects. Such an environment can hinder productivity, slow down development processes, and complicate the overall management of tasks.

Lumis addresses this challenge by offering a unified platform that consolidates essential features into one seamless environment. With advanced automated SQL generation and extraction, developers can manage databases with ease, streamlining data retrieval and manipulation processes. Furthermore, Lumis incorporates text generation and summarization capabilities, allowing users to create and refine written content quickly and efficiently. By leveraging the Gemini API and the LangChain framework, Lumis provides powerful tools for YouTube video transcription and website transcription, enabling users to convert audio and web content into editable text formats. The platform also supports multiformat file retrieval and aggregation, making it easier for users to manage diverse sources of information in one place.

A standout feature of Lumis is its innovative code assistance powered by the TAPAS model. This feature transforms the debugging experience by allowing developers to pinpoint errors on their screens using their mouse, verbally describe the issue, and receive immediate, context-aware solutions. This interactive troubleshooting method significantly reduces the time spent on debugging, enhancing overall efficiency in the development process. Additionally, the platform includes Air Canva, a tool similar to Canva, which empowers users to create visually appealing content effortlessly.

In summary, Lumis is designed to be a comprehensive, intelligent platform that integrates critical development and content creation tools—SQL management, text generation, summarization, transcription services, and advanced code assistance—into one unified system. By focusing on continuous updates and providing a user-friendly interface, Lumis ensures that developers and content creators can work more effectively and efficiently, adapting to the increasingly complex demands of their projects. This holistic approach not only simplifies workflows but also enhances productivity, making Lumis a vital resource for anyone looking to thrive in the modern software development landscape.

1.2 Problem Statement

The software development ecosystem currently lacks a comprehensive unified platform that seamlessly integrates essential functionalities, such as real-time code assistance, text generation, content creation, and efficient workflow management. Developers often struggle with a fragmented array of tools that do not provide timely feedback or adapt to the evolving needs of modern software development, highlighting the urgent necessity for a holistic solution that encompasses all these features.

1. **Unified Platform for All Features** : Developers frequently face challenges due to the reliance on multiple disconnected tools for different tasks, including coding, SQL management, and content creation. This fragmentation leads to inefficiencies as they must transition between various platforms to complete their work, resulting in wasted time and increased cognitive strain. A cohesive platform that consolidates crucial features—such as text generation, retrieval-augmented generation (RAG), SQL capabilities, and transcription services—into one unified environment is essential. Such integration would enable developers to manage all aspects of their workflow seamlessly, enhancing overall efficiency and productivity.

2. **Real-Time Code Assistant Utilizing Mouse Pinpointing and User Audio** : Existing code assistance tools often require manual input and lack the contextual awareness needed for effective debugging. A real-time code assistant that leverages mouse pinpointing allows developers to visually identify errors on their screens while describing issues verbally. This innovative approach not only improves the debugging experience by delivering immediate, context-aware feedback but also significantly reduces the time spent on troubleshooting. By combining visual cues with audio input, this solution provides a more interactive and efficient means for developers to manage code errors, ultimately boosting productivity.

3. **Engaging Feature Using Air Canva and Automated SQL Generation and Extraction** : Current content creation tools can be complex and often lack the flexibility needed for dynamic design tasks. Integrating an engaging feature like Air Canva into the platform enables users to create visually compelling content with ease. This functionality empowers both developers and content creators to design graphics and presentations without requiring advanced design skills. Additionally, the inclusion of automated SQL generation and extraction simplifies database interactions, allowing users to generate and retrieve queries effortlessly. By making the content creation process more intuitive and enjoyable, users can produce high-quality visuals while managing their data effectively, thereby streamlining the overall development workflow.

4. **Dynamic and User-Friendly Interface** : Many existing development tools are hindered by complex and unintuitive interfaces, which can impede user adoption and reduce productivity, particularly for less experienced developers. A dynamic, user-friendly interface is crucial for making the platform accessible and easy to navigate. Such an interface would cater to users of varying skill levels and improve the overall user experience, allowing developers to concentrate on their tasks rather than struggling with convoluted tools. By emphasizing usability and interactivity, the platform can cultivate a more efficient and satisfying development environment.

1.3 Objectives

The objectives of the **LUMIS** project outline a comprehensive approach to enhancing coding and application development through the integration of advanced technologies. By focusing on real-time error detection, multimodal functionalities, automated SQL processes, and seamless technological integration, **LUMIS** aims to create a unified platform that significantly improves productivity and streamlines workflows.

1. Real-Time Code Error Detection and Resolution
LUMIS aims to develop a multimodal system that detects and resolves code errors in real time using video and audio inputs. Users can highlight errors on their screens with mouse pinpointing while describing issues through audio, ensuring immediate feedback. This dual input method reduces debugging time and minimizes disruptions, ultimately enhancing productivity.

2. Integration of Advanced Functionalities
Another key objective is to integrate functionalities like text generation, retrieval-augmented generation (RAG), and transcription into a cohesive platform. Text generation enables users to create high-quality written content, while RAG enhances information retrieval for informed decision-making. Transcription services convert audio and video content into editable text, streamlining information management and providing a powerful toolset for users.

3. Automated SQL Generation and Data Extraction
LUMIS will facilitate automated SQL generation and data extraction, improving database management efficiency. Leveraging Generative AI (GenAI), the system can generate SQL queries based on user needs, saving time and minimizing errors. Integration with OpenCV and Air Canvas technology allows for real-time problem-solving based on user-drawn annotations, enriching the user experience and simplifying data management.

4. Seamless Integration and Continuous Updates
LUMIS is committed to seamless integration of technologies and continuous updates to maintain functionality and relevance. This ensures the platform adapts to evolving user needs and technological advancements. Regular updates enhance usability and keep **LUMIS** user-friendly, providing a reliable and efficient tool that evolves alongside user requirements.

1.4 Scope

The scope of the **LUMIS** project outlines the key functionalities and features that will be developed to create a unified multimodal intelligent system. By focusing on integrated error detection, interactive debugging tools, unified AI capabilities, and an advanced user interface, **LUMIS** aims to deliver an efficient and productive experience for users.

1. Integrated Error Detection
LUMIS will develop a system that combines video, audio, and text inputs for the instant identification and resolution of coding errors. This multimodal approach ensures that developers receive immediate, context-aware feedback, significantly enhancing the efficiency of error detection and correction.

2. Interactive Debugging Tools
The project will create functionalities that enable users to actively pinpoint and correct errors on-screen. With features that provide immediate feedback, developers can interactively engage with their code, facilitating a more efficient debugging process and reducing time spent on troubleshooting.

3. Unified AI Features
LUMIS aims to merge advanced AI capabilities into a single, efficient platform. This includes text generation for creating high-quality written content, retrieval-augmented generation (RAG) for enhanced information retrieval, and transcription services for converting audio and video into editable text. The integration of these functionalities will provide users with a comprehensive toolset that streamlines their workflows.

4. Advanced User Interface and Automation
The project will focus on designing a dynamic, user-centric interface that enhances the overall user experience. Additionally, **LUMIS** will automate SQL generation and data management processes to facilitate seamless interaction with databases, ultimately improving productivity and simplifying complex tasks for users.

Chapter 2

Literature Review

The rise of artificial intelligence (AI) has brought transformative advancements across numerous sectors, with significant potential to revolutionize multimodal interactions and automate complex tasks. Large language models (LLMs), such as OpenAI and Gemini, have been at the forefront of these advancements, providing unprecedented capabilities in understanding and processing different forms of data. These models are now being integrated into unified systems capable of handling a wide array of inputs, including text, images, and videos, to deliver a more streamlined user experience. Traditional approaches to multimodal processing typically involve separate tools for each type of input, which often results in fragmented workflows, inefficiencies, and increased complexity for users. By unifying these functionalities into one cohesive platform, advanced systems aim to address these challenges, providing a seamless experience that simplifies task execution and minimizes errors. This evolution represents a significant shift in how multimodal data is processed and managed, with applications in various fields such as content creation, data analysis, and software development. This literature review explores the application of advanced technologies in developing intelligent, multimodal systems that can perform a wide range of tasks. It will also examine the challenges associated with integrating these technologies into user-centric AI platforms, including the complexities of ensuring real-time performance, adaptability to diverse user needs, and the ethical considerations of AI-driven automation. Furthermore, the review will highlight the opportunities these systems present in driving innovation and filling gaps in current research, offering insights into how such technologies can be optimized for broader and more efficient use.

2.1 Comparative Analysis of Recent Study

The comparative analysis of recent studies highlights key advancements in language models and code understanding. Patrick Lewis et al. (2020) [4] present a retrieval-augmented generation model that enhances language generation but faces high computational costs and reliance on external knowledge. Iqbal H. Sarker (2024) [8]emphasizes challenges related to misinformation and ethical concerns in LLMs, while Sumit Kumar Dam et al. (2024) [3] provide an overview lacking depth on ethical implications. Yanwei Li et al. (2024) [5]introduce Mini-Gemini for Vision Language Models, criticized for its resource intensity. Furthermore, Pooyan Rahmanzadehgervi et al. (2024) [7] demonstrate that VLMs struggle with basic visual tasks. Overall, these studies underscore the pressing need for improved interpretability, scalability, and ethical considerations in AI deployment.considerations in AI deployment.

Sr. No	Title	Author(s)	Year	Methodology	Drawback
1	Retrieval Augmented Generation for Knowledge-Intensive NLP task [4]	Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, Douwe Kiela	2020	The methodology combines seq2seq models with Dense Passage Retrieval (DPR) to enhance language generation. It uses RAG-Token for autoregressive beam search and RAG-Sequence with thorough or fast decoding for efficient text generation.	The RAG model relies heavily on external knowledge (Wikipedia), which limits scope and incurs high computational costs for retrieval. It reduces but doesn't eliminate hallucinations, lacks full interpretability, and struggles with real-time updates or multimodal inputs.
2	LLM potentiality and awareness: a position paper from the perspective of trustworthy and responsible AI modeling [8]	Iqbal H. Sarker	2024	The methodology includes defining tasks and acquiring data, selecting and fine-tuning models, and evaluating their performance with metrics. Post-evaluation, models are deployed and monitored to ensure reliability.	The paper highlights LLMs' susceptibility to generating misinformation, biases, and adversarial attacks, while also being opaque in decision-making and raising privacy and ethical concerns. These issues challenge trust, fairness, and responsible deployment in AI systems.
3	A Complete Survey on LLM-based AI Chatbots [3]	Sumit Kumar Dam, Choong Seon Hong, Yu Qiao, Chaoning Zhang	2024	The paper surveys the evolution, applications, and challenges of LLM-based chatbots, from early models to current systems like ChatGPT. It aims to provide a comprehensive overview and explore future improvements for these technologies.	This paper does not adequately address specific challenges faced by LLM-based chatbots, such as ethical concerns and data biases, nor does it discuss technical limitations like scalability and complex conversation handling. It lacks concrete examples or solutions related to the misuse of generated knowledge.

Table 2.1: Comparative Analysis of Literature Survey

Sr. No	Title	Author(s)	Year	Methodology	Drawback
4	Mini-Gemini: Mining the Potential of Multi-modality Vision Language Models [5]	Yanwei Li1*, Yuechen Zhang1*, Chengyao Wang1*, Zhisheng Zhong1, Yixin Chen1, Ruihang Chu1, Shaoteng Liu1, Jiava Jia1,	2024	The methodology of Mini-Gemini focuses on improving Vision Language Models (VLMs) by enhancing visual tokens through an additional visual encoder for high-resolution refinement, constructing a high-quality dataset that supports precise image comprehension and reasoning-based generation, and leveraging VLM-guided generation techniques.	Mini-Gemini's emphasis on high-resolution visual tokens, complex architecture, and computational demands makes it overly resource-intensive and ill-suited for audio-centric projects like yours, where simpler and more focused frameworks such as Whisper and Sentence-BERT are more effective.
5	Using an LLM to Help With Code Understanding [6]	Daye Nam, Andrew Macvean, Vincent Hellendoorn, Bogdan Vasilescu, Brad Myers	2024	A user study with 32 participants compared GILT to web search, analyzing task completion and code comprehension. Quantitative and qualitative data were collected.	No significant improvement in task completion time or understanding. Benefits varied between students and professionals.
6	Creating Large Language Model Applications Utilizing LangChain: A Primer on Developing LLM Apps Fast [9]	Oguzhan Topsakal, and Tahir Cetin Akinci	2023	LangChain, a framework designed to build AI applications utilizing large language models (LLMs). It uses modular components like prompts, chains, and memory to develop customizable pipelines that interact with external data sources for tasks like question-answering and document summarization.	LangChain's reliance on LLMs means it may still face common issues like generating inaccurate outputs. Additionally, handling complex tasks can require multiple API calls, which may lead to slower execution or increased computational costs.

Table 2.2: Comparative Analysis of Literature Survey

Sr. No	Title	Author(s)	Year	Methodology	Drawback
7	Gemini in Reasoning: Unveiling Commonsense in Multimodal Large Language Models [10]	Yuqing Wang, Yun Zhao	2023	Four LLMs and two MLLMs are tested on 12 commonsense reasoning datasets using zero-shot and few-shot learning to assess accuracy, with focus on visual-textual integration.	The study is limited to English datasets, missing cross-cultural nuances, and results may vary with future model updates or new datasets.
8	Vision language models are blind [7]	Pooyan Rahmanzadehgervi, Logan Bolton, Mohammad Reza Taesiri, Anh Totti Nguyen	2024	The BlindTest benchmark evaluates four top Vision Language Models (VLMs) on seven low-level visual tasks, such as counting shapes and identifying intersections, focusing on simple geometric challenges.	VLMs perform poorly on basic visual tasks, with an average accuracy of 58.57.
9	Unit Test Generation using Generative AI: A Comparative Performance Analysis of Autogeneration Tools [2]	Shreya Bhatia, Tarushi Gandhi, Dhruv Kumar Pankaj Jalote	2024	Compared unit test generation tools using Python code samples categorized by structure. Evaluated tools on 109 Python projects, excluding NumPy-dependent ones. Selected core modules based on LOC, complexity, and import dependencies.	Limited by Pynguin's inability to handle NumPy projects, few large samples in Category 1, and potential bias in file selection.
10	Few-shot training LLMs for project-specific code-summarization [1]	Toufique Ahmed, Premkumar Devanbu	2022	The methodology - used Codex for few-shot code summarization. Structured prompts with code-summary pairs and a target function. Evaluated on CodeXGLUE using 1,000 samples due to limited Codex access.	Limited access to the Codex model constrained the dataset size and overall experimentation. The 4000-token limit for few-shot training restricted the number of sequences, and using too much data could lead to catastrophic forgetting. Zero-shot and one-shot training provided lower performance compared to few-shot methods.

Table 2.3: Comparative Analysis of Literature Survey

Chapter 3

Project Design

3.1 Proposed System Architecture

The Lumis architecture is a versatile system designed to accommodate multiple forms of input, making it a comprehensive solution for multimodal data processing. It accepts Finger Notations, Image Input, Text Input, Screen Audio Recordings, Mouse Input, Multiple Format Files, and MySQL Data. These varied inputs are processed by specialized modules to deliver precise and meaningful results. The AIR Canva Module handles image interpretation and finger notations, while the Text Generation & Summarization Module manages natural language processing, transforming text into concise summaries.

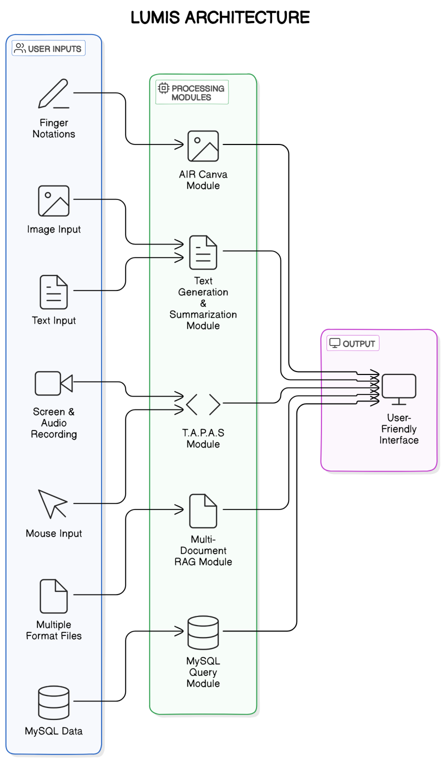


Figure 3.1: Proposed System Architecture of Lumis

The T.A.P.A.S Module processes and analyzes screen and audio recordings, converting them into actionable insights through transcription or analysis. The Multi-Document RAG Module efficiently retrieves and generates content from multiple documents, ensuring data accuracy and relevance. Additionally, the MySQL Query Module enables robust database querying. All these modules feed their outputs into a user-friendly interface, providing seamless interaction and easy access to complex data insights. By organizing tasks into specific processing modules, Lumis offers a highly adaptable, modular, and user-centric platform capable of handling a wide range of tasks across various data formats.

3.2 Data Flow Diagrams (DFD)

The data processing flowchart outlines how various input types are processed through specialized modules to generate meaningful outputs, enhancing user experience and productivity. It begins with users uploading image and text inputs, which are sent to the Text Generation & Summarization Module. This module analyzes the content to produce concise summaries or newly generated text tailored to user needs, which is then delivered back to them. Concurrently, screen and audio recordings, along with mouse inputs, are directed to the T.A.P.A.S Module, focusing on real-time error detection and debugging, providing immediate feedback and solutions to streamline problem-solving and workflow.

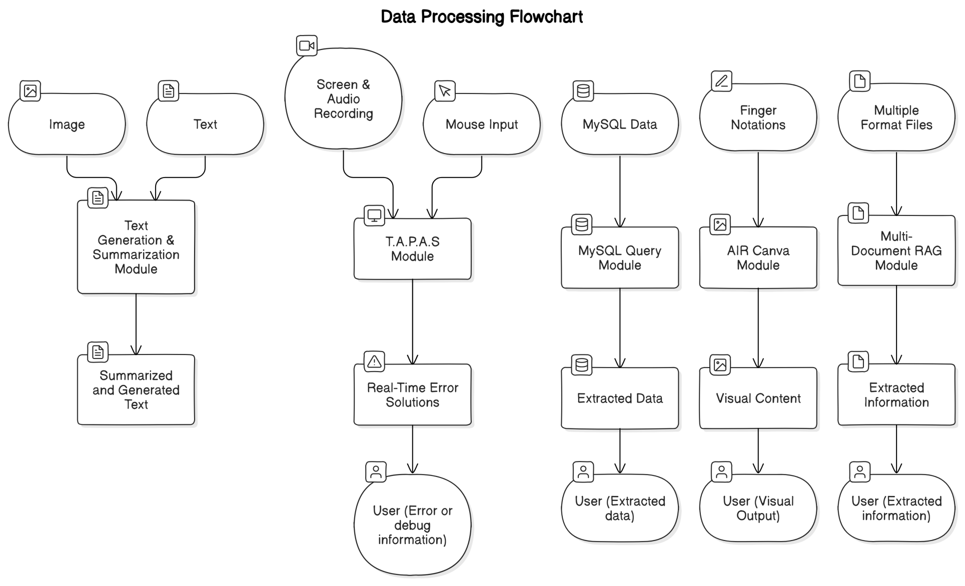


Figure 3.2: Data Flow Diagram of Lumis

The MySQL Query Module handles data retrieval, allowing users to execute queries that extract organized information from the database for easy access. Additionally, user-provided finger notations are processed by the AIR Canva Module, transforming gestures into visually appealing content to enhance engagement. The Multi-Document RAG Module processes multiple format files, extracting valuable information from various documents and presenting it clearly to users. Together, these modules convert diverse input types into practical outputs—whether in the form of text summaries, real-time error solutions, engaging visuals, or structured data—significantly improving user interaction and productivity.

3.3 Use Case Diagrams

The use case diagram outlines the interactions between Users and the System Admin within the LUMIS system. Users engage with various features such as Transcription, Image-to-Text Conversion, and Tapas Model Assistance for AI-driven tasks. They also utilize Real-time Code Assistance, the Air Canva Feature for content creation, and automate Text Generation and SQL Management, allowing them to efficiently handle complex tasks.

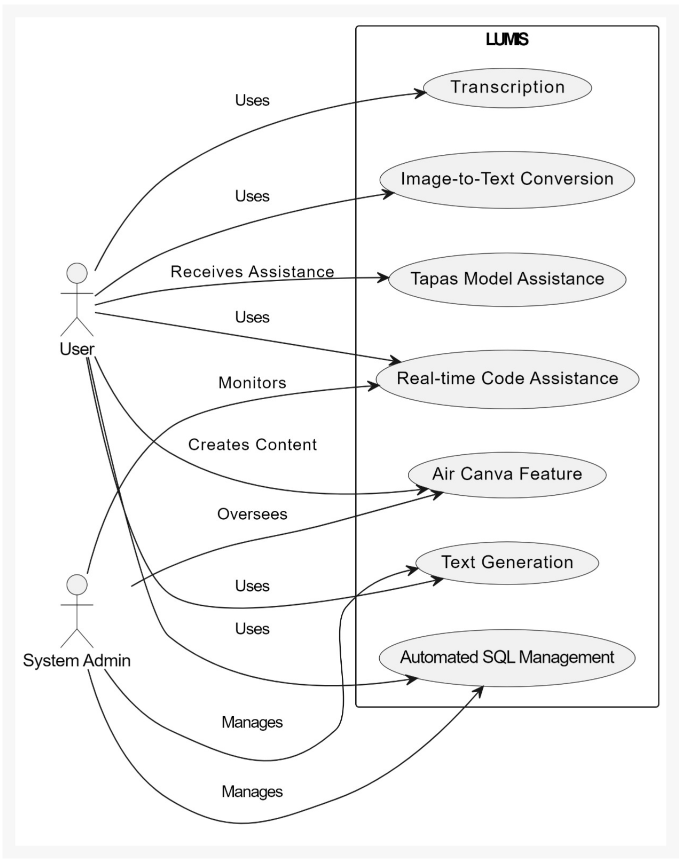


Figure 3.3: Use Case Diagram of Lumis

Meanwhile, the System Admin oversees key functionalities, managing transcription services, image-to-text conversion, and SQL management while monitoring user content creation activities. This collaboration between Users and the System Admin enhances productivity and streamlines workflows, making the LUMIS platform more efficient and user-friendly.

Chapter 4

Project Implementation

The implementation phase of the LUMIS project focused on transforming the planned design and architecture into a fully functional and scalable system. This phase involved building essential features such as real-time code assistance, text generation, image-to-text conversion, and SQL management. The backend development included integrating multiple APIs, including OpenAI and Gemini, to power core functionalities, while the user interface was developed using Django and React for a smooth and responsive user experience. The integration of the Tapas model for dynamic code error solutions was a key focus, ensuring real-time assistance with visual error tracking. Throughout this phase, rigorous testing of individual components and code snippets was conducted to ensure functionality, robustness, and security. Additionally, the system's scalability was optimized to handle high data loads and user demands. Automated documentation generation and analytics tracking were incorporated to monitor user interactions and system performance, enabling continuous improvement. Ultimately, the implementation phase successfully brought together all technical elements into a unified, working solution that is ready for testing and deployment.

4.1 Timeline Sem VII

As Project Lumis grew in complexity, well-structured project management and milestone tracking became essential. This timeline highlights key phases during Semester VII, tracking the project's evolution from conceptualization to design, implementation, and testing. Each stage was meticulously planned to ensure efficiency, mitigate risks, and maintain alignment with the project's goals.

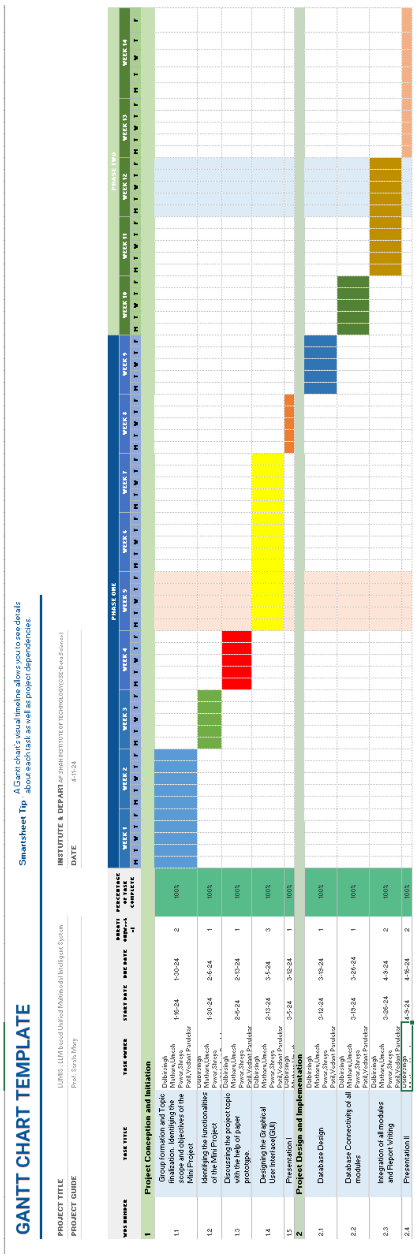
The project involved numerous sophisticated tasks, requiring strong collaboration between team members and adherence to deadlines. The Gantt chart (Figure 4.1) visually illustrates the timeline, showing how each activity contributed to the overall workflow and how task dependencies influenced project progression. This was critical for coordinating overlapping tasks and avoiding bottlenecks in a complex development workflow.

In the early phases, the team defined the project scope and researched methodologies that would underpin the multimodal system. Integration of APIs such as OpenAI and Gemini enabled key functionalities, including text generation, image-to-text conversion, YouTube transcription, website transcription, and multiformat file retrieval, establishing the core of Lumis's intelligent capabilities.

As the project moved into design and implementation, attention turned to developing real-time code error detection, intelligent suggestions, and multimodal inputs, all integrated with popular code editors to boost developer productivity. Advanced features like Air Canva, voice assistance, and SQL integration were also developed to enhance functionality and user experience.

Throughout the semester, regular reviews and progress checks ensured that the project stayed on track. Adjustments were made as necessary to address task prioritization and overcome challenges, such as managing multiple concurrent tasks. The Gantt chart (Figure 4.1) also tracked task completion percentages, providing a clear view of progress at a glance.

Overall, this timeline serves as a comprehensive overview of Project Lumis's lifecycle during Semester VII, showcasing how the team managed its complexities, adhered to deadlines, and ensured smooth task completion through strategic planning and collaboration.



PROJECT TITLE	LUMIS : LLM based Unified Multimodal Intelligent System
PROJECT GUIDE	Prof. Sarala Mary
INSTUTUTE & DEPART	AP SHAH INSTITUTE OF TECHNOLOGY(CSE-Data Science)
DATE	4-11-24

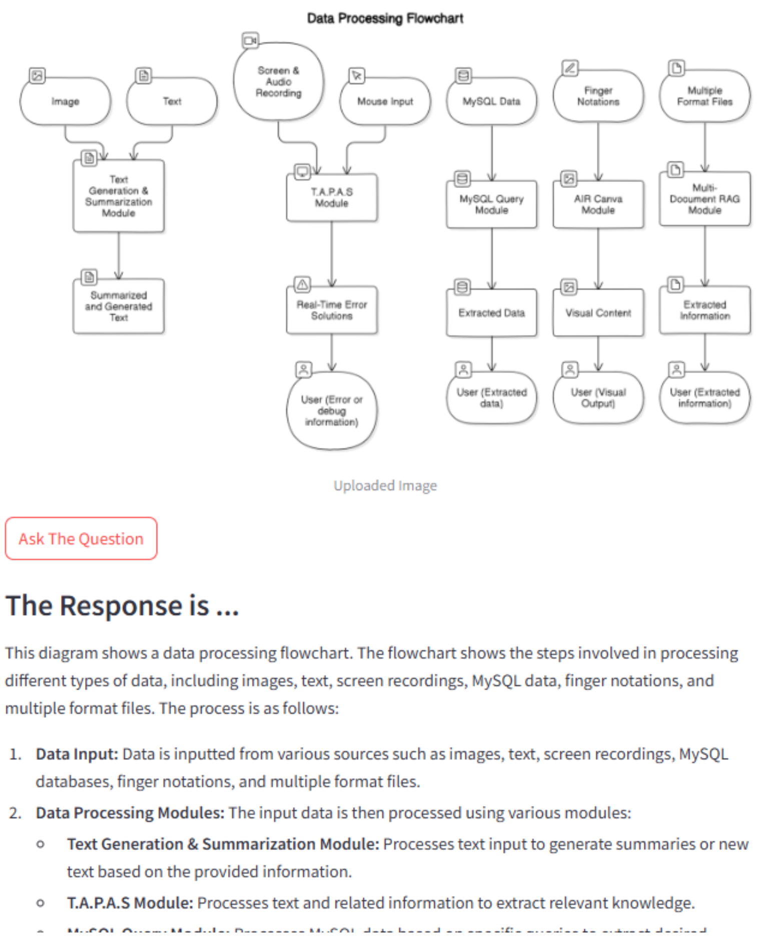
WBS NUMBER	TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATI ON(Week)	PERCENTAGE OF TASK COMPLETE
1	Project Conception and Initiation					
1.1	Group formation and Topic finalization. Identifying the scope and objectives of the Mini Project	Dalbirsingh Matharu,Umesh Pawar,Shreya Patil,Vedant Parulekar	1-16-24	1-30-24	2	100%
1.2	Identifying the functionalities of the Mini Project	Dalbirsingh Matharu,Umesh Pawar,Shreya	1-30-24	2-6-24	1	100%
1.3	Discussing the project topic with the help of paper prototype.	Dalbirsingh Matharu,Umesh Pawar,Shreya Patil,Vedant Parulekar	2-6-24	2-13-24	1	100%
1.4	Designing the Graphical User Interface(GUI)	Dalbirsingh Matharu,Umesh Pawar,Shreya Patil,Vedant Parulekar	2-13-24	3-5-24	3	100%
1.5	Presentation I	Dalbirsingh	3-5-24	3-12-24	1	100%
2	Project Design and Implementation					
2.1	Database Design	Dalbirsingh Matharu,Umesh Pawar,Shreya Patil,Vedant Parulekar	3-12-24	3-19-24	1	100%
2.2	Database Connectivity of all modules	Dalbirsingh Matharu,Umesh Pawar,Shreya Patil,Vedant Parulekar	3-19-24	3-26-24	1	100%
2.3	Integration of all modules and Report Writing	Dalbirsingh Matharu,Umesh Pawar,Shreya Patil,Vedant Parulekar	3-26-24	4-9-24	2	100%
2.4	Presentation II	Dalbirsingh	4-9-24	4-16-24	2	100%

Figure 4.1: Timeline of Project

4.2 System Prototype

In this section, we present the system prototype developed for our Gemini-powered document summarization application. The prototype serves as a tangible representation of our envisioned solution, integrating the capabilities of the Gemini API to process and summarize text from various document formats, particularly PDFs. By leveraging a user-friendly interface the prototype allows users to easily upload files, receive instant summaries, and interact with the underlying technology. This section will detail the architecture, features, and functionalities of the prototype, highlighting its design choices and the rationale behind them. The objective is to demonstrate the potential of the system in real-world scenarios, paving the way for future enhancements and deployments. The image in the provided Figure 4.1

is processed using an image-to-text extraction technique, where the content of the flowchart is analyzed and converted into readable text. In this process, an AI model—powered by Google's Gemini API—was employed to recognize the image's contents and interpret the various components and steps outlined in the diagram. The flowchart illustrates a data processing system with several inputs, such as text, images, and recordings, which are processed through different modules like the Text Generation Summarization Module, the MySQL Query Module, and the Multi-Document Retrieval-Augmented Generation (RAG) Module.



The Response is ...

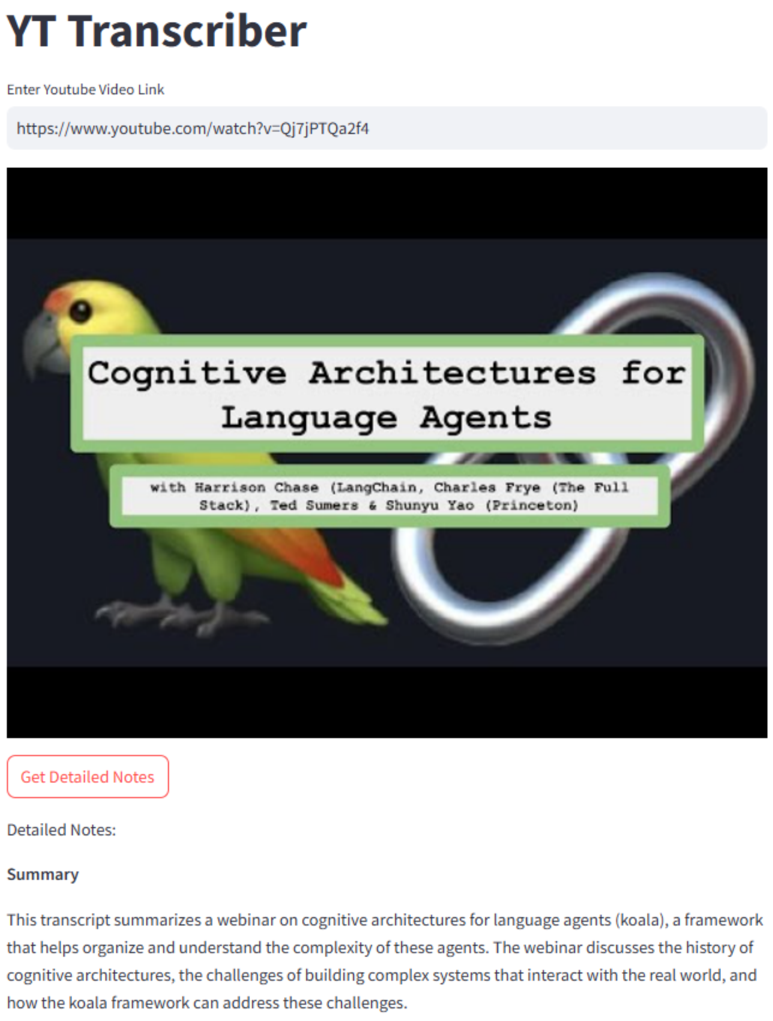
This diagram shows a data processing flowchart. The flowchart shows the steps involved in processing different types of data, including images, text, screen recordings, MySQL data, finger notations, and multiple format files. The process is as follows:

1. **Data Input:** Data is inputted from various sources such as images, text, screen recordings, MySQL databases, finger notations, and multiple format files.
2. **Data Processing Modules:** The input data is then processed using various modules:
   - **Text Generation & Summarization Module:** Processes text input to generate summaries or new text based on the provided information.
   - **T.A.P.A.S Module:** Processes text and related information to extract relevant knowledge.
   - **MySQL Query Module:** Processes MySQL data based on specific queries to extract desired

Figure 4.2: Image To Text

These modules handle data processing to generate outputs like summarized text, visual content, and extracted data. The process begins with various input types and ends with user-facing results or error/debugging information. The image processing technique extracts the corresponding text and structure from the flowchart, making the content more accessible for further analysis or interaction within the application.

In the Figure 4.2, a YouTube video link is entered into a transcriber interface, and the system processes this video using a transcription service powered by Google's Gemini API. This involves extracting the audio content from the YouTube video and converting it into a text transcript. The specific video titled "Cognitive Architectures for Language Agents" is transcribed, and a summary of the video is generated.


YT Transcriber

Enter Youtube Video Link

https://www.youtube.com/watch?v=Qj7jPTQa2f4

Cognitive Architectures for Language Agents
with Harrison Chase (LangChain, Charles Frye (The Full Stack), Ted Sumers & Shunyu Yao (Princeton)
Get Detailed Notes

Detailed Notes:

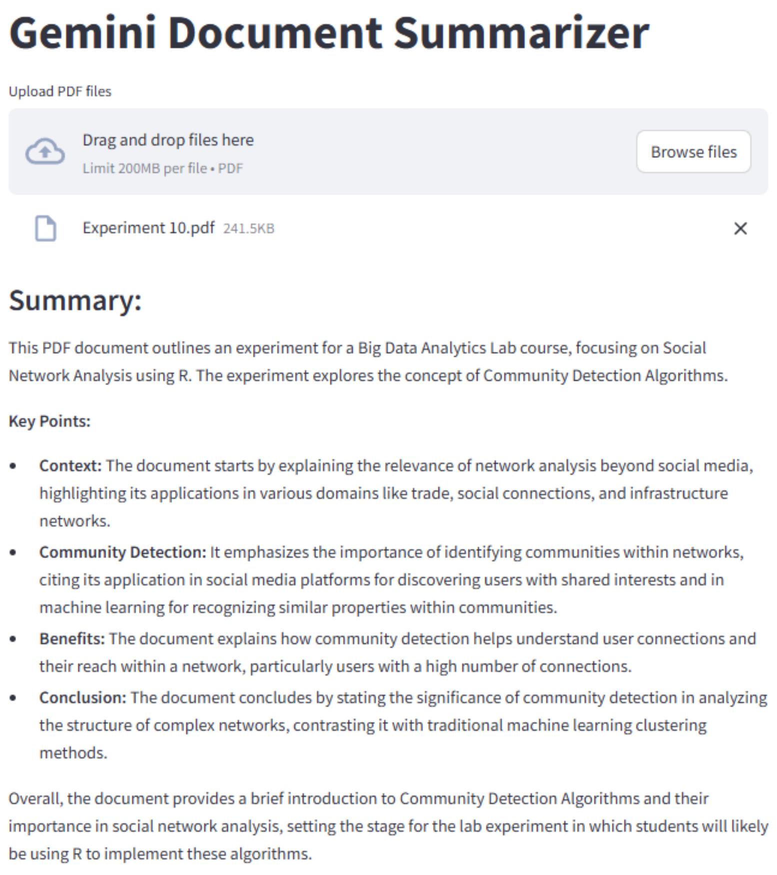
Summary

This transcript summarizes a webinar on cognitive architectures for language agents (koala), a framework that helps organize and understand the complexity of these agents. The webinar discusses the history of cognitive architectures, the challenges of building complex systems that interact with the real world, and how the koala framework can address these challenges.

Figure 4.3: YouTube Transcription

The summary highlights a discussion on cognitive architectures, with a focus on the Koala framework and the challenges of designing complex systems that interact with the real world. The Gemini API plays a crucial role in automatically analyzing and summarizing the video content, making it easier for users to access the key points without needing to watch the full video.

The Figure 4.3 illustrates the Gemini Document Summarizer, a tool that leverages a Gemini API key to enable users to upload PDF files and receive concise summaries of their content. Users can easily drag and drop PDFs or browse to upload, with a maximum file size limit of 200MB. Once a file is uploaded, the tool processes the document and generates a structured summary, highlighting key points and offering a clear breakdown of the main ideas.



Gemini Document Summarizer

Upload PDF files

Drag and drop files here
Limit 200MB per file • PDF

Browse files

Experiment 10.pdf 241.5KB

Summary:

This PDF document outlines an experiment for a Big Data Analytics Lab course, focusing on Social Network Analysis using R. The experiment explores the concept of Community Detection Algorithms.

Key Points:

- **Context:** The document starts by explaining the relevance of network analysis beyond social media, highlighting its applications in various domains like trade, social connections, and infrastructure networks.
- **Community Detection:** It emphasizes the importance of identifying communities within networks, citing its application in social media platforms for discovering users with shared interests and in machine learning for recognizing similar properties within communities.
- **Benefits:** The document explains how community detection helps understand user connections and their reach within a network, particularly users with a high number of connections.
- **Conclusion:** The document concludes by stating the significance of community detection in analyzing the structure of complex networks, contrasting it with traditional machine learning clustering methods.

Overall, the document provides a brief introduction to Community Detection Algorithms and their importance in social network analysis, setting the stage for the lab experiment in which students will likely be using R to implement these algorithms.

Figure 4.4: Multiple PDFs RAG

Additionally, users have the option to remove uploaded files if necessary, further enhancing the tool's usability and making it a convenient solution for summarizing large documents efficiently.

Chapter 5

Summary

Project Lumis is an innovative initiative that seeks to transform the software development landscape through the creation of an integrated multimodal intelligent system powered by advanced large language models (LLMs). The primary aim of Lumis is to address the fragmentation of tools that developers often face by providing a unified platform that consolidates essential functionalities. This comprehensive system is designed to facilitate text generation, transcription services, image-to-text conversion, and automated SQL management, enabling users to efficiently manage their tasks and enhance their productivity.

At the core of Lumis is a suite of intelligent AI agents developed using the LangChain framework. This allows the platform to function as an adaptive assistant, learning from user interactions to make informed decisions and automate complex tasks. Notably, the real-time code assistant feature offers immediate feedback on coding errors, significantly reducing the time spent on debugging and allowing developers to maintain focus on their primary tasks. Furthermore, the integration of SQL functionalities streamlines database management, enabling efficient query generation and data manipulation. These features collectively empower developers to navigate the complexities of their projects with confidence and precision.

In addition to core functionalities, Lumis incorporates tools that enhance user engagement and creativity. The inclusion of Air Canva, a design feature similar to popular graphic design platforms, enables users to create visually appealing content with ease. This is particularly beneficial for content creators who require high-quality visuals but may lack advanced design skills. By focusing on user-friendliness and an intuitive interface, Lumis caters to a diverse user base, ensuring that both novice and experienced developers can leverage its capabilities without unnecessary hurdles.

In conclusion, Project Lumis stands as a comprehensive solution that integrates vital functionalities to streamline workflows and enhance productivity in the modern software development environment. By providing a unified platform that combines text generation, transcription, image processing, real-time code assistance, and SQL management, Lumis addresses the multifaceted challenges faced by developers and content creators. Its commitment to continuous improvement and user-centric design ensures that it remains relevant and effective in an ever-evolving technological landscape. Ultimately, Lumis empowers users to work more effectively, adapt to complex demands, and thrive in their respective fields, marking a significant advancement in the realm of software development tools.

Bibliography

[1] Toufique Ahmed and Premkumar Devanbu. Few-shot training llms for project-specific code-summarization. In *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*, pages 1–5, 2022.

[2] Shreya Bhatia, Tarushi Gandhi, Dhruv Kumar, and Pankaj Jalote. Unit test generation using generative ai: A comparative performance analysis of autogeneration tools. In *Proceedings of the 1st International Workshop on Large Language Models for Code*, pages 54–61, 2024.

[3] Sumit Kumar Dam, Choong Seon Hong, Yu Qiao, and Chaoning Zhang. A complete survey on llm-based ai chatbots. *arXiv preprint arXiv:2406.16937*, 2024.

[4] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

[5] Yanwei Li, Yuechen Zhang, Chengyao Wang, Zhisheng Zhong, Yixin Chen, Ruihang Chu, Shaoteng Liu, and Jiaya Jia. Mini-gemini: Mining the potential of multi-modality vision language models. *arXiv preprint arXiv:2403.18814*, 2024.

[6] Daye Nam, Andrew Macvean, Vincent Hellendoorn, Bogdan Vasilescu, and Brad Myers. Using an llm to help with code understanding. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, pages 1–13, 2024.

[7] Pooyan Rahmanzadehgervi, Logan Bolton, Mohammad Reza Taesiri, and Anh Totti Nguyen. Vision language models are blind. *arXiv preprint arXiv:2407.06581*, 2024.

[8] Iqbal H Sarker. Llm potentiality and awareness: a position paper from the perspective of trustworthy and responsible ai modeling. *Discover Artificial Intelligence*, 4(1):40, 2024.

[9] Oguzhan Topsakal and Tahir Cetin Akinci. Creating large language model applications utilizing langchain: A primer on developing llm apps fast. In *International Conference on Applied Engineering and Natural Sciences*, volume 1, pages 1050–1056, 2023.

[10] Yuqing Wang and Yun Zhao. Gemini in reasoning: Unveiling commonsense in multi-modal large language models. *arXiv preprint arXiv:2312.17661*, 2023.