Smart Lecture Assistant: A Multimodal AI Framework for Automated Lecture Transcription, Summarization, and Assessment Generation

[Project Report Submitted to Central University of South Bihar & NIELIT, Patna in Partial Fulfilment of The Requirement of The Degree of Master in Data Science & Applied Statistics]

By

NAIMISH PADHAN

CUSB2302222004

Masters in Data Science & Applied Statistics

[2023-2025]

Under Supervision of

Mr. Ankit Kumar

Scientist C, NIELIT, Patna

DEPARTMENT OF STATISTICS

CENTRAL UNIVERSITY OF SOUTH BIHAR, GAYA

DECLARATION

The work embodied in the project entitled as "**Smart Lecture Assistant: A Multimodal AI Framework for Automated Lecture Transcription, Summarization, and Assessment Generation**" Was initiated at the Department of Statistics under School of Mathematics, Statistics, and Computer Science, Central University of South Bihar, Gaya, Bihar, in partial fulfilment of the requirement for the award of Master's in Data Science & Applied Statistics. This work has not been submitted in part or full, to this or any other university or institution, for any degree or diploma.

DATE……. **Signature**

(NAIMISH PADHAN)

दक्षिण बिहार केन्द्रीय विश्वविद्यालय
Central University of South Bihar
Department of Statistics
School of Mathematics, Statistics, and Computer Science
SH-7, Gaya Panchanpur Road, Village - Karhara, Post. Fatehpur, Gaya-824236
(BIHAR) Phone/Fax :0631-2229530 2229514, Website:www.cush.ac.in

CERTIFICATE

This is to certify that his project entitled **"Smart Lecture Assistant: A Multimodal AI Framework for Automated Lecture Transcription, Summarization, and Assessment Generation"** is submitted by NAIMISH PADHAN, enrolment number CUSB2302222004, in partial fulfilment for award of Master's in Data Science & Applied Statistics of Central University of South Bihar. This work has not been submitted for the award of any degree/diploma of this or any other university and it is his original work.

Date……...

Signature

(Dr. Sunit Kumar)

Head of Department

National Institute of Electronics and Information Technology, Patna

Ministry of Electronics & Information Technology, Government of India

राष्ट्रीय इलेक्ट्रॉनिक की एवं सूचना प्रौद्योगिकी संस्थान, पटना

रा.इ.सू.प्रौ.सं NIELIT

CERTIFICATE

This is to certify that Naimish Padhan (CUSB2302222004), pursuing Master in Data Science & Applied Statistics at the Central University of South Bihar, Gaya has worked at NIELIT, Patna under the supervision of Mr. Ankit Kumar Scientist C, NIELIT, Patna on project entitled **"Smart Lecture Assistant: A Multimodal AI Framework for Automated Lecture Transcription, Summarization, and Assessment Generation"** .

Date……...

Signature

(Mr. Ankit Kumar)

Supervisor

Scientist C, NIELIT, Patna

ACKNOWLEDGEMENT

This success of this work would have been impossible without the guidance of my supervisors and mentors; help form friends and support from my family. I would like to extend my appreciation to all of them who have been a part of this.

At the outset, special appreciation goes to my supervisor, Mr. Ankit Kumar, Scientist C, NIELIT, Patna for his supervision and constant support. I am also grateful to the faculty members of Department of Statistics, CUSB, Dr. Sunit Kumar (HOD), Dr. Indrajeet Kumar, Dr. Sandeep Kumar Maury, Dr. Kamlesh Kumar, for their help and support. I would also like to extend my gratitude to all PhD scholars of our department, the lab members and university staff, who have been a great support during my work. Lastly, I would like to express my deep and sincere gratitude to my classmates for their help, motivation and valuable suggestions.

NAIMISH PADHAN

TABLE OF CONTENT

Chapter-1

Chapter-2

Chapter-3

Chapter-4

Chapter-5

Chapter-6

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

In the age of digital education, the demand for intelligent tools that assist in efficient learning and teaching is rapidly increasing. This thesis presents Smart Lecture Assistant: A Multimodal AI Framework for Automated Lecture Transcription, Summarization, and Assessment Generation, a comprehensive system that leverages state-of-the-art artificial intelligence to enhance lecture content accessibility and comprehension. The proposed system integrates speech recognition, natural language processing, and machine translation to automatically process recorded lectures or uploaded educational materials (videos or PDFs).

At its core, the system employs Whisper, an open-source speech-to-text model developed by OpenAI, to transcribe spoken lectures into accurate textual form. This transcription is optionally passed through the Google Translate API to support multilingual users. The transcribed or translated text is then processed by Gemini Pro, a powerful language model, to generate human-like summaries and multiple-choice questions (MCQs). The system also extracts mathematical equations. Finally, the results are delivered in user-friendly formats, including downloadable PDFs and TXT files, through an interactive Flask-based web application.

This multimodal framework addresses the critical challenges of manual note-taking, content revision, and personalized assessment in education. Experimental evaluations demonstrate the system's effectiveness in producing coherent summaries, relevant assessments, and accurate transcriptions. The thesis concludes with potential enhancements, including OCR support for handwritten notes and real-time classroom integration, envisioning a future where AI-driven educational assistants become integral to modern pedagogy.

CHAPTER-1

1. INTRODUCTION

1.1 . Background and Motivation

Today's educational world looks nothing like it did just a few years ago. Students are watching recorded lectures on their phones during commutes, attending virtual seminars from their living rooms, and accessing course materials that span continents. This digital revolution has opened incredible doors for learning, but it's also created some real challenges that anyone who's tried to make sense of a three-hour recorded lecture can relate to.

Picture this: you're a student trying to review for an exam, faced with hours of recorded content, dense PDF documents, and webinar recordings that seem to go on forever. Or maybe you're an instructor who wants to help your students but finds yourself spending countless hours manually creating study guides and summaries. The truth is, while we have more educational content available than ever before, much of it sits there like an unopened treasure chest—valuable but difficult to access efficiently.

The problem becomes even more complex when we consider technical subjects filled with specialized terminology, mathematical concepts, or content delivered in languages that aren't everyone's first language. Students often find themselves drowning in information rather than swimming through knowledge, spending more time trying to organize and understand the format of their materials than actually learning from them.

This is where the idea for our Smart Lecture Assistant was born. We recognized that the same artificial intelligence technologies transforming other industries could revolutionize how people interact with educational content. Why should students spend hours transcribing lectures when AI can do it instantly? Why should educators manually create summaries when intelligent systems can extract the most important points automatically?

Our solution brings together some of the most powerful AI tools available today: Whisper transforms spoken words into accurate text, Gemini creates meaningful summaries and generates thoughtful assessments, and Google Translate breaks down language barriers that might prevent someone from accessing quality education. But this isn't just about combining cool technologies—it's about creating something that genuinely makes learning more accessible, efficient, and enjoyable.

The real driving force behind this project is deeply human. We want to give students back their time so they can focus on understanding concepts rather than organizing content. We want to support teachers who are already doing incredible work by giving them tools that amplify their impact. And perhaps most importantly, we want to ensure that language barriers, learning differences, or simply the overwhelming amount of available content never stand between someone and the education they deserve.

This Smart Lecture Assistant represents our vision of how AI can transform passive content consumption into active, engaging learning experiences. It's not about replacing human connection in education—it's about enhancing it, making it more accessible, and helping both learners and educators focus on what they do best: thinking, creating, and growing together.

1.2 . Problem Statement

Let's be honest about what's happening in classrooms and study spaces around the world right now. Students are sitting in front of their laptops at 2 AM, trying to scrub through a two-hour recorded lecture to find that one crucial concept their professor mentioned somewhere in the middle. They're frantically taking notes while watching videos at 1.5x speed, hoping they don't miss anything important. Meanwhile, educators are spending their weekends manually creating study materials and assessments, wishing there was a better way to help their students engage with the content they've worked so hard to deliver.

This isn't just inconvenient—it's a real barrier to learning. When students have to spend more time wrestling with the format and organization of their educational materials than actually learning from them, something's broken. A typical lecture recording might contain gold mines of information, but if a student can't efficiently extract what they need for their upcoming exam or project, that information might as well not exist.

The situation gets even more frustrating when you consider what students actually need: they want quick access to key points, they need ways to test their understanding, and they often require content in their preferred language or format. What they get instead are long, unstructured video files with no easy way to navigate, review, or interact with the material meaningfully.

Current solutions feel like they were designed by people who never actually had to study from recorded lectures. You might find a decent transcription tool here, a basic summarization service there, but nothing that works together seamlessly. These isolated tools often miss the nuances of educational content—they stumble over technical terminology, completely butcher mathematical concepts, and treat a physics lecture the same way they'd treat a casual conversation. For international students or those who speak multiple languages, the situation is even more challenging, as most tools simply don't account for the diverse, multilingual nature of today's educational landscape.

What we're really missing is a system that understands education. We need something that can listen to a lecture and not just write down what was said, but actually understand what matters most. We need tools that can create meaningful summaries that capture the essence of complex topics, generate questions that actually test understanding rather than just memory, and make all of this accessible to learners regardless of their language background or learning preferences.

The gap we're addressing isn't just technological—it's deeply human. Students need more time to think, explore, and understand rather than transcribe and organize. Educators need

tools that amplify their teaching rather than add to their workload. And everyone in the educational ecosystem needs better ways to make learning more accessible, engaging, and effective.

This is exactly why we've developed the "Smart Lecture Assistant"**.** We're not just combining different AI technologies for the sake of it—we're creating an integrated solution that understands the real challenges people face when trying to learn from digital content. By bringing together Whisper's accurate transcription capabilities, Gemini's intelligent summarization and assessment creation, and Google Translates multilingual support, we're building something that actually serves the people who will use it: students trying to learn efficiently and educators trying to teach effectively.

Our goal is simple but ambitious: to transform how people interact with educational content, making it more accessible, more engaging, and ultimately more effective for learning. We want to give students back their time and give educators tools that enhance rather than complicate their important work.

1.3 . Objectives

Our mission with the **Smart Lecture Assistant** is straightforward yet ambitious: we want to create a tool that genuinely makes learning from digital content easier, more engaging, and accessible to everyone. Rather than forcing students and educators to adapt to technology, we're building technology that adapts to how people actually learn and teach.

Here's what we're working toward:

i) **Making Spoken Words Crystal Clear** We're harnessing Whisper, one of the most sophisticated speech recognition systems available, to transform any lecture audio or video into accurate, readable text. This isn't just about getting the words right—it's about creating transcriptions that actually make sense, capture the flow of ideas, and handle the technical language that's so common in educational content. Whether it's a chemistry professor explaining molecular structures or a literature teacher discussing narrative techniques, our system needs to get it right the first time.

ii) **Turning Information Overload into Clear Insights** Using Google Gemini's powerful language understanding, we're building a summarization system that doesn't just shorten content—it distils it into the essence of what students need to know. Think of it as having a brilliant study partner who can listen to any lecture and pull out the key concepts, main arguments, and critical details that matter most for understanding and retention.

iii) **Creating Meaningful Ways to Test Understanding** Learning isn't just about consuming information; it's about being able to use and apply that knowledge. Our system generates multiple-choice questions that go beyond simple recall to test real comprehension. These aren't random quiz questions—they're carefully crafted assessments that help students identify what they've mastered and where they need to focus more attention.

iv) **Breaking Down Language Barriers** Education shouldn't be limited by language. By integrating Google Translate, we're ensuring that valuable educational content can reach learners regardless of their native language. Whether someone needs their engineering lecture translated from English to Spanish, or wants to access content originally delivered in another language, our system makes knowledge truly global and accessible.

v) **Building Something People Actually Want to Use** The best technology in the world is useless if it's too complicated or frustrating to use. We're creating a Flask-based web application that feels intuitive and welcoming—somewhere students can simply upload their lecture videos or PDF materials and get exactly what they need without having to learn a new system or navigate confusing interfaces.

vi) **Proving It Actually Works** We're not just building something that looks impressive in a demo. We're committed to rigorously testing every component of our system—measuring how accurate our transcriptions are, how helpful our summaries prove to be, how effective our assessment questions are at testing real understanding, and how well our translations preserve meaning. Most importantly, we're listening to the people who use our system to understand what works and what needs improvement.

vii) **Contributing Something Meaningful to Education** Beyond solving immediate problems, we want to demonstrate something important: that thoughtfully combining different AI technologies can create solutions that are genuinely transformative for education. We're not just building another educational tool—we're showing how AI can enhance the human elements of teaching and learning rather than replacing them.

Every objective we've set reflects our core belief that technology should serve people, not the other way around. We're building the Smart Lecture Assistant because we believe that every student deserves efficient access to quality education, every educator deserves tools that amplify their impact, and every person should be able to learn in the way that works best for them, regardless of language, location, or learning style.

This isn't just about creating a successful project—it's about contributing to a future where artificial intelligence makes education more human, more accessible, and more effective for everyone involved.

1.4 . Scope of the Study

Let's talk about what we've built and be upfront about where we're focusing our efforts. The Smart Lecture Assistant is designed to tackle the everyday challenges that students and educators face when dealing with digital lecture content, and we've been strategic about what we're including in this version.

Here's What We're Covering:

i) **Turning Speech into Text That Actually Makes Sense** Using Whisper**,** we can take any lecture video or audio recording and transform it into clean, readable text. This isn't just basic transcription—we're talking about understanding context,

handling academic terminology, and producing something you'd actually want to read and study from. Right now, we're focusing primarily on English-language lectures, though we're building the foundation for expanding to other languages down the road.

ii) **Making Long Lectures Digestible** Whether you're working with a transcribed video lecture or uploading PDF lecture notes, our system uses Google Gemini to create summaries that actually capture what matters. We're not just cutting content down to size—we're identifying the key concepts, main arguments, and essential information that students need to focus on for effective learning.

iii) **Creating Study Questions That Matter** The system generates multiple-choice questions directly from lecture content, giving students a way to test their understanding and identify knowledge gaps. These questions are designed to go beyond simple memorization, though we'll be honest—we're still working on capturing the deepest levels of conceptual reasoning that the best human-created assessments achieve.

iv) **Breaking Down Language Barriers** Through Google Translate, we're making transcriptions and summaries available in multiple languages. This opens up educational content to international students, multilingual learners, and anyone who might benefit from accessing material in their preferred language.

v) **Handling Different Types of Content,** we've built the system to work with the materials educators and students actually use: video and audio lecture recordings for spoken content, and PDF documents for written lecture notes, slides, and handouts. This flexibility means you can use our tool regardless of how your educational content is delivered.

vi) **Making It Easy to Use** Our **Flask-based web application** is designed with real users in mind. Upload your files, get your results, and download everything in the formats you need—whether that's simple text files for quick reference or PDFs for more formal documentation. No complicated setup, no confusing interfaces.

vii) **Focusing on Real Educational Settings** We're specifically targeting higher education environments, online learning platforms, and academic content delivery systems—places where automated processing can genuinely enhance learning rather than just add unnecessary complexity.

viii) **Let's Be Clear About What We're Not Doing (Yet)** We believe in being transparent about our current limitations. This version doesn't support real-time streaming—you'll need to upload complete files rather than process live lectures as they happen. Our summarization and question generation work best with English content right now, and while our MCQs are solid for testing factual understanding, they're not yet sophisticated enough to assess the kind of deep conceptual reasoning that advanced learners need to demonstrate.

ix) **Why These Boundaries Matter** Rather than trying to solve every possible educational technology challenge at once, we've focused on creating a tool that does specific things really well. We'd rather have a system that excellently handles the core challenges of transcription, summarization, assessment, and translation than one that tries to do everything but doesn't do any of it particularly well.

This focused approach means we can deliver genuine value to students and educators right now, while building a foundation that can grow and expand as we learn more about how people actually use these tools in their educational practice. We're not just building technology for technology's sake—we're creating solutions for real problems that real people face every day in their learning and teaching.

1.5. Significance of the Study

The development of the Smart Lecture Assistant holds significant value in the evolving landscape of education and artificial intelligence. As educational institutions increasingly adopt digital learning environments, there is a growing need for tools that can enhance comprehension, accessibility, and engagement with academic content. This study contributes to that need through an AI-driven framework that automates lecture transcription, summarization, and assessment creation.

The key contributions and significance of the study are as follows:

i) **Enhancing Learning Efficiency**

By providing concise summaries and automated assessments, the system allows students to focus on key concepts without having to revisit lengthy lectures or extensive documents. This improves retention and optimizes study time.

ii) **Bridging Accessibility Gaps**

With integrated speech-to-text and translation features, the system supports diverse learners, including those with hearing impairments and non-native language speakers, making educational content more inclusive.

iii) **Automating Educator Workflows**

The assistant reduces the manual effort educators invest in preparing summaries and quizzes, allowing them to focus on higher-level teaching strategies and student engagement.

iv) **Multimodal and Scalable Framework**

By supporting both video/audio and PDF inputs, the framework adapts to various content formats, making it useful for universities, MOOCs, and self-paced learners alike.

v) **Application of Advanced AI Models**

The study integrates state-of-the-art models such as Whisper for transcription and Gemini Pro for summarization and question generation. This practical application showcases the potential of generative AI in real-world educational tools.

vi) **Contribution to EdTech Research**

The project serves as a case study in combining multimodal AI technologies within a single framework, providing a foundation for future research and development in educational technology.

vii) **Foundation for Personalized Learning**

The assistant opens possibilities for further customization, such as generating topic-specific summaries, adaptive quizzes, or learning analytics, thereby paving the way for personalized and data-driven education.

CHAPTER-2

2. LITRATURE REVIEW

2.1. Lecture Capture and Educational technology

The adoption of lecture capture systems has revolutionized education by offering students flexible access to learning content. These systems typically involve the recording, storage, and retrieval of lecture in digital formats. According to studies, such tools improve learning outcomes, especially for visual and auditory learners. However, traditional systems often lack advanced features like automatic summarization or assessment generation, leaving significant scope for intelligent AI-powered enhancements. This gap motivates the integration of multimodal AI techniques into modern educational tools.

2.2. Automatic Speech Recognition (ASR) – Overview of whisper

Automatic Speech Recognition (ASR) refers to the process of converting spoken language into text. OpenAI's Whisper, a state-of-the-art ASR model, is trained on 680,000 hours of multilingual and multitask supervised data.it supports robust transcription across a wide range of languages and accents, including noisy and real-world environments. Whisper uses an encoder-decoder transformer architecture and shows competitive performance, especially in zero-short and multilingual scenarios. This makes it highly suitable for lecture transcription where clarity, domain-specific vocabulary, and spontaneous speech are common challenges.

2.3. Text Summarization Techniques – Extractive vs Abstractive

Text summarization is an essential NLP task that involves condensing long pieces of text into shorter versions while preserving essential information.

There are two primary approaches:

- Extractive Summarization: Selects key sentences directly from the source. Traditional algorithms include TextRank, LSA, LexRank.

- Abstractive Summarization: Generates new sentences that paraphrase the content. Deep learning models like BERTSum, T5, Pegasus, and Gemini Flash fall into this category.

Abstractive methods, like those used by Google's Gemini, better mimic human-like summaries and are ideal for dynamic content such as lectures.

2.4. Question Generation in NLP

Question Generation is a critical component of educational AI. It involves the automatic creation of relevant questions from a given passage. Classical approaches relied on rule-based systems and syntactic parsing. Modern neural Question Generation models leverage sequence-to-sequence architectures with attention mechanisms, fine-tuned on datasets like SQuAD, SciQ, and QuAC. Gemini's advanced capabilities allow it to understand context and generate Multiple Choice Questions that align with Bloom's Taxonomy, covering knowledge, comprehension, and application-level skills.

2.5. Machine Translation – Role of Google Translate

Machine Translation (MT) enables cross-linguistic access to content. Google Translate, one of the most widely used MT tools, employs deep learning-based Neural Machine Translation (NMT) models. It uses encoder-decoder LSTM or transformer architectures to produce fluent and context-aware translations. Despite being a general-purpose translator, it performs well for academic language when content is pre-processed properly. This facilitates content accessibility for multilingual learners in educational platforms.

2.6. Review of Related Work

P. Pawar, Z. Gulhane, R. Dube, R. Patil, and A. Joshi (2024) presented a research paper on "Automated Generation and Evaluation of Multiple-Choice Quizzes using Langchain and Gemini LLM" which demonstrates the application of advanced AI technologies for educational content creation. Their work leverages Google's Gemini AI model and Langchain framework to automatically generate contextually relevant multiple-choice questions from PDF documents, with customizable difficulty levels and subject-specific content generation [1]. This study is highly relevant to Smart

Lecture Assistant projects as it showcases how Large Language Models can be integrated into educational tools to automate assessment creation, provide intelligent content processing from educational materials, and offer adaptive learning experiences. The research demonstrates key capabilities that align with smart lecture systems: automated content extraction from documents, intelligent question generation based on complexity levels, real-time evaluation and feedback mechanisms, and user-friendly interfaces for educational applications. Their methodology of using prompt engineering, sequential chaining of AI models, and comprehensive evaluation metrics provides a foundation for developing intelligent educational assistants that can process lecture materials and generate interactive learning content automatically.

R. J. Paul, S. Jamal, S. Bejoy, R. J. Daniel, and N. Aju (2024) presented a research paper on "QGen: Automated Question Paper Generator" which demonstrates the application of fine-tuned T5 transformer models for automated educational content generation from PDF documents. Their work utilizes natural language processing techniques to extract text from educational materials and generate contextually relevant questions, achieving 70% relevance accuracy through human evaluation [2]. This study is directly relevant to Smart Lecture Assistant projects as it showcases the practical implementation of transformer-based models for processing educational documents and generating interactive assessment content. The research demonstrates key functionalities essential for intelligent lecture systems: automated text extraction and preprocessing from PDF materials, fine-tuned question generation using transfer learning approaches, web-based user interface for faculty interaction, and document storage and retrieval systems using MongoDB. Their methodology of segmenting extracted text, fine-tuning T5 models on educational datasets like SQuAD, and providing PDF output generation capabilities offers valuable insights for developing smart lecture assistants that can process lecture materials and create automated assessments. The system's ability to save and retrieve previously generated content aligns with the persistent storage requirements of intelligent educational platforms that need to maintain lecture history and generated materials for future reference.

H. Gaur, D.K. Tayal, and A. Jain (2023) presented a research paper where they developed ViQG (Viva Question Generator), a web tool for automatic question generation from programming code to assist students in viva preparation. "ViQG Web Tool for Automatic Question Generation from Code for Viva Preparation" [3]. In this study, they proposed the QGCode method that processes programming code in C, Java, and Python languages to extract keywords and automatically generate relevant viva questions. The system uses keyword matching with a pre-built dataset containing 150, 200, and 100 viva questions for C, Java, and Python respectively. Their approach demonstrates how Natural Language Processing techniques can be applied to enhance student learning through automated question generation, providing a student-centric solution rather than traditional teacher-centric applications.

CHAPTER-3

3. SYSTEM ARCHITECTURE AND DESIGN

3.1. System Overview

The Smart Lecture Assistant is an intelligent, multimodal AI-powered web application designed to streamline the educational content creation process by automating lecture transcription, summarization, translation, and assessment generation. It is built to support students and educators by converting raw lecture materials (videos and PDFs) into structured, summarized content accompanied by meaningful assessments.

At its core, the system leverages state-of-the-art pre-trained models and APIs to handle various stages of the pipeline:

- Whisper: An automatic speech recognition (ASR) model developed by OpenAI is used to transcribe spoken content from lecture videos into text.
- Google Translate: The transcribed content can be translated into English or another target language to support multilingual accessibility.
- Gemini Model (Google AI): This model generates comprehensive summaries, multiple-choice questions (MCQs), and optional concept maps from the extracted lecture content.

The application is deployed via a Flask-based web interface that allows users to upload lecture materials, view real-time summaries, and download outputs such as:

- Summarized lecture notes
- MCQs for self-assessment

This system aims to enhance the learning experience by reducing the manual workload involved in note-taking and quiz preparation while making lectures more accessible and engaging.

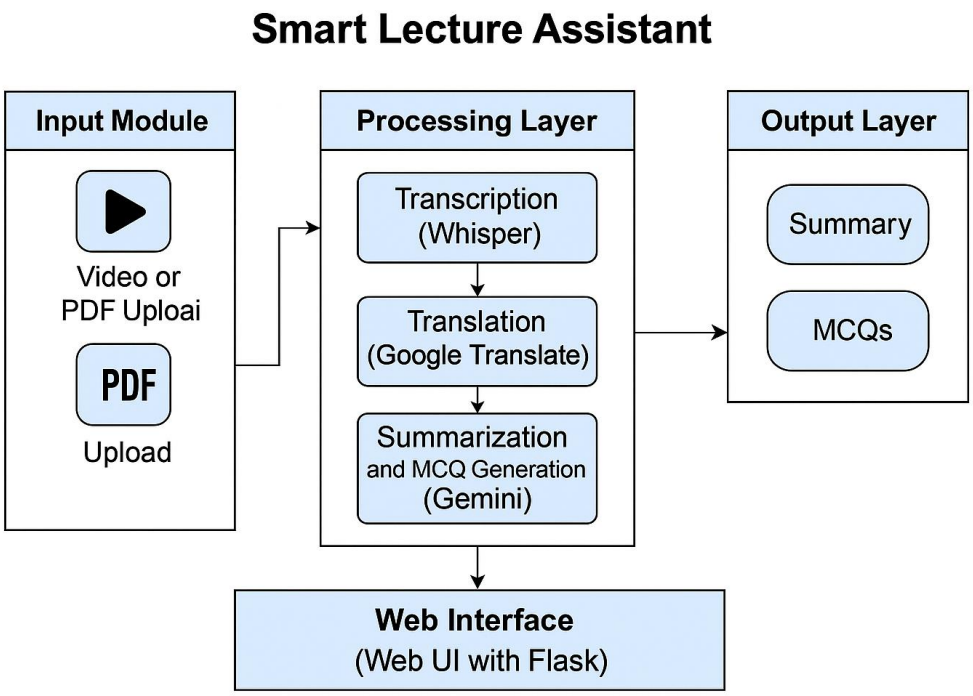


Figure-3.1: Overall System Workflow

3.2. Workflow Pipeline

The Smart Lecture Assistant operates through a structured workflow pipeline, enabling seamless processing of video or PDF content into valuable learning materials. The system is modular and follows a sequential architecture, ensuring each phase contributes to the final output effectively.

I. Input Module (Video/PDF Upload)

This module allows users to upload lecture materials in two formats:

- Video files: Typically, MP4 or similar formats containing spoken content.
- PDF files: Lecture slides, notes, or handouts.

The input files are validated for format and integrity before processing.

II. Transcription with Whisper

Once a video is uploaded, Whisper, a state-of-the-art open-source automatic speech recognition (ASR) model developed by OpenAI, is employed to:

- Extract and transcribe spoken content into text.
- Maintain temporal alignment with the original audio.
- Handle multiple accents and background noise with high accuracy.

3.2.1. Input Module (Video/PDF Upload)

The input module is responsible for accepting lecture materials from users.

It supports:

- Video Upload: Common formats like MP4, MOV, AVI.
- PDF Upload: Lecture slides or notes in PDF format.

Upon upload, the system detects the file type and routes it to the appropriate handler:

- Video – Whisper model for audio transcription.
- PDF – Text extractor for direct processing.

3.2.2. Transcription with Whisper

The Whisper model, developed by OpenAI, is used to convert audio from video into text using Automatic Speech Recognition. It is capable of handling noisy environments and multiple languages.

Key Features:

- Multilingual ASR
- Timestamped transcription
- Robust to scents and low-quality audio

The transcribed text is then passed to the translation module if the language is not English.

3.2.3. Translation with Google Translate

To ensure compatibility with downstream NLP models, all content must be in English. The Google Translate API is used to:

- Automatically detect the language of transcription.
- Translate to English if needed.

3.2.4. Summarization using Gemini Model

Once the lecture text is in English, it is sent to the Gemini Flash model to generate an accurate and concise summary of the lecture content.

Gemini's Role:

- Capture key points and themes
- Eliminate redundancies
- Handle technical and academic language effectively

The output is structured in paragraph from and optimized for clarity and readability.

3.2.5. MCQ Generation

Based on the summarized content, Gemini is again utilized to create Multiple Choice Questions. This supports self-assessment and educational reinforcement.

Features:

- Unique questions with one correct answer and three distractors
- Context-based generation from summary
- Can include math and concept-based questions

The generated MCQs are formatted and presented through the output interface.

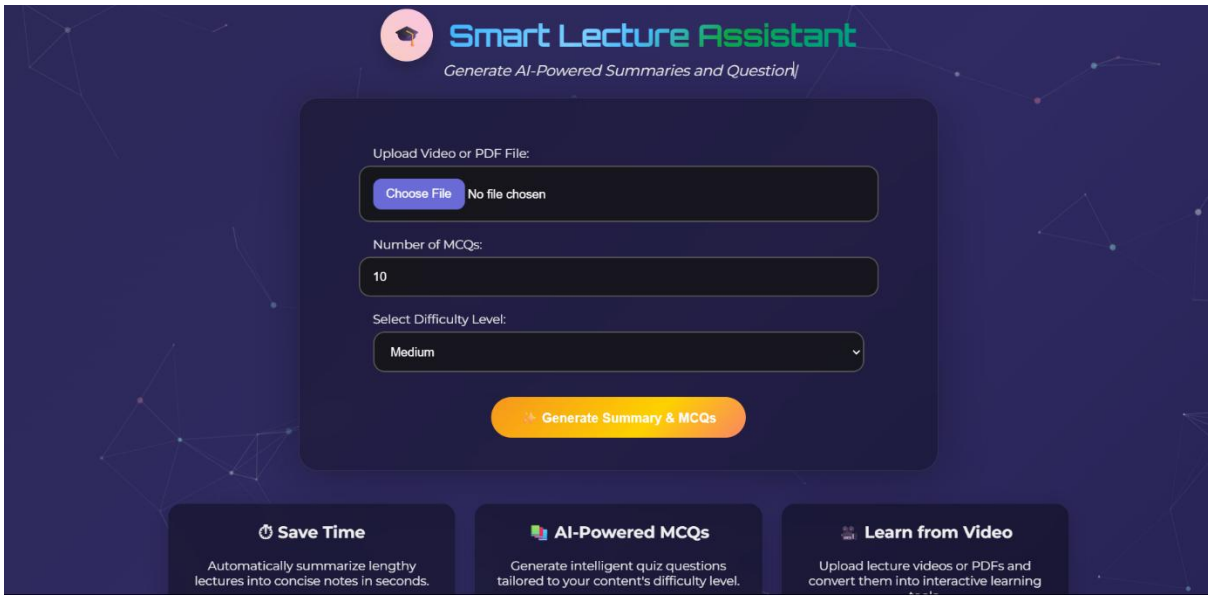
3.3. User Interface Design (Web UI with Flask)

The Smart Lecture Assistant offers a clean, intuitive, and responsive web interface built using Flask combined with HTML, CSS, and JavaScript. The design focuses on user experience, accessibility, and functionality for academic users such as students, educators, and researchers.

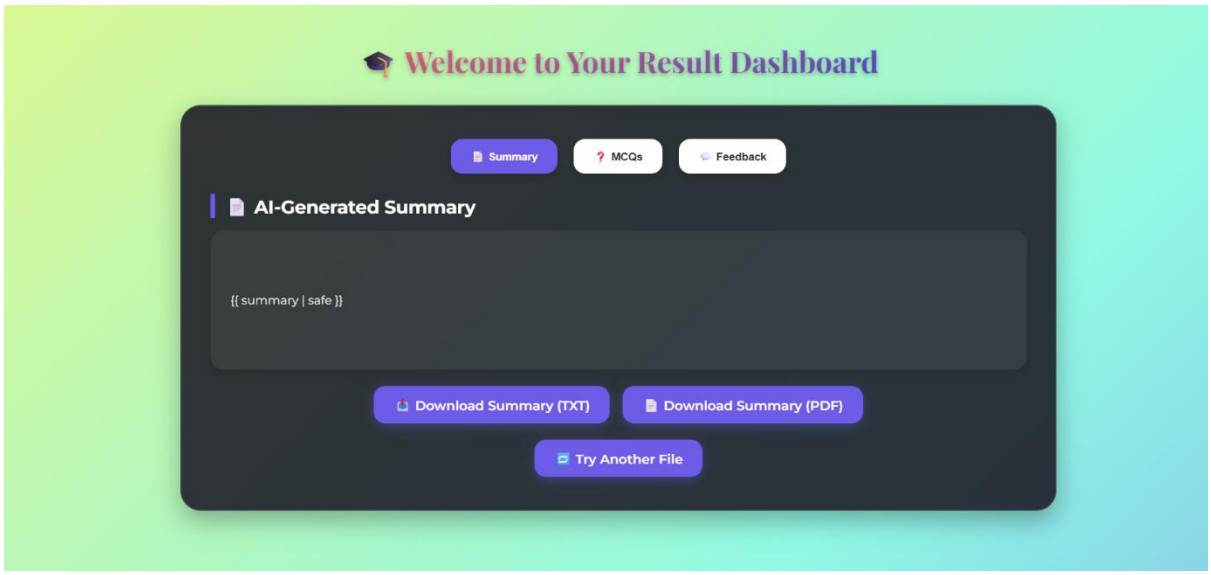
Key UI features:

- File upload interface: Manual file upload options for both video and PDFs.
- Progress Feedback: Real-time loading animation and progress indicators while processing files.
- Result Preview: Users can preview the summary, MCQs before downloading.
- Download Option: Results can be downloaded in .txt or .pdf format

Index Page:



Result Page:



3.4. Tools and Technologies Used

The project incorporates a range of modern technologies to implement the multimodal AI pipeline and deliver a robust, interactive platform.

Component	**Technology/Tool**	**Purpose**
Programming Language	Python	Backend logic, API integration
Framework	Flask	Web server and routing
Frontend	HTML, CSS, JavaScript	UI/UX design
Transcription Engine	Whisper (OpenAI)	Audio-to-text conversion
Translation Service	Google Translate API	Language translation
NLP & AI APIs	Gemini Flash (Google AI Studio)	Summarization, MCQ generation
PDF Handling	PyMuPDF, FPDF	Text extraction and PDF output
Audio Processing	MoviePy, ffmpeg	Extracting audio from video
Text Formatting	Markdown, Unicode	Rendering math and clean formatting
Version Control	Git, GitHub	Code management and collaboration
Deployment Environment	Localhost	Execution and testing

Table-3.1: Tools and Technologies Used

CHAPTER-4

4. METHODOLOGY

4.1. Dataset Description (Videos, PDFs)

The Smart Lecture Assistant system is designed to operate on two primary types of educational content: lecture videos and PDF documents. Instead of relying on a static dataset, the system dynamically processes real-time user inputs, making it flexible and scalable for various academic environments.

4.1.1. Lecture Videos

Lecture videos from the core modality for spoken content and are typically source from:

- **Academic Institutions:** NPTEL, MIT OpenCourceWare, Coursera, EdX.
- **User-Generated Content:** Classroom recordings, instructor-led tutorials, seminar captures.
- **Video Specifications:**
    - Formats: .mp4, .avi, .mov, .mkv
    - Duration: Typically, 10-60 minutes
    - Language: Multi-language support with a focus on English and Hindi

Purpose in system:

- Used for automated speech recognition (ASR) via the Whisper model.
- Generates the base transcription for summarization and assessment.

4.1.2. Lecture PDFs

PDF files represent structured written content such as lecture notes, research papers, presentation slides, and textbooks.

- Source:
  - Uploaded by students or teachers.
  - Exported from academic learning management systems (LSM) or Google Slides.
- Document Types:
  - Lecture Slides
  - Study Guides
- PDF Processing Goals:
  - Extract coherent and ordered text.
  - Preserve mathematical formulas and key concepts.
  - Enable AI-driven summarization and MCQ generation.

4.1.3. Dataset Flexibility and Design Rationale

Modality	Format	Purpose	Processed By
Lecture Video	.mp4, .avi	Speech transcription	Whisper
Lecture PDFs	.pdf	Text summarization, MCQ creation	PyMuPDF + Gemini

Table-4.1: Data Flexibility and Design Rational

4.2. Preprocessing Techniques

Preprocessing plays a critical role in ensuring that the input data—whether from videos or PDF files—is clean, coherent, and optimized for downstream AI tasks like summarization, translation, and question generation. Since the Smart Lecture Assistant processes two types of data modalities (audio/video and text), different preprocessing strategies are used for each.

4.2.1. Preprocessing for Lecture Videos

Lecture videos undergo audio-focused preprocessing before they can be transcribe using the Whisper model.

Steps Involved:

- Audio Extraction:
  - The video file is processed using tools like ffmpeg to extract the audio in a compatible format (.wav or .mp3).
- Audio Normalization:
  - The audio is normalized to a uniform volume level to improve transcription accuracy.
- Noise Reduction:
  - Background noise, echoes, or silences are minimized using audio filters or preprocessing libraries such as pydub.
- Segmentation:
  - Long audio files may be split into shorter segments to enhance processing speed and transcription consistency.

4.2.2. Preprocessing for PDF Documents

PDF documents require a different set of preprocessing steps, focusing on structured text extraction and context organization.

Steps Involves:

- PDF Parsing:
  - Libraries like PyMuPDF or pdfminer.six are used to extract raw text from PDFs while attempting to retain structure.
- Noise Removal:
  - Removal of:
    - Headers/footers
    - Page numbers
    - Watermarks
    - Repeated irrelevant phrases

- Whitespace and Formatting Normalization:
    - Correcting improper line breaks, extra whitespaces, or special characters that could confuse the summarization model.
- Equation handling:
    - Mathematical equations are preserved using Unicode math symbols.

4.2.3. Language Detection & Translation

To support multilingual content:

- A language detection module is applied to identify the content's original language.
- If not in English, it is translated using the Google Translate API before feeding it into the Gemini summarization engine.

4.2.4. Tokenization and Chunking

- For long transcripts or PDFs, the content is broken down into manageable "chunks" to respect into token limit of models like Gemini.
- Text chunks are tokenized using the SentencePiece tokenizer or BART tokenizer equivalents.
- Chunk size is dynamically calculated to fit within API limit while maintaining context.

4.2.5. Summary of Preprocessing Pipeline

Input Type	**Preprocessing Steps**
Video/Audio	Audio extraction – Normalization – Noise reduction – Segmentation - Whisper
PDF	Text extraction -Cleanup – Structure preservation - Tokenization
Any Text	Language detection – Optional translation – Chunking – Passed to Gemini

Table-4.2: Summary of Preprocessing pipeline

4.3. Whisper: Speech-to-Text Architecture and integration

Whisper, developed by OpenAI, is an advanced automatic speech recognition (ASR) system trained on a large-scale dataset of multilingual and multitask audio. In this project, Whisper plays a central role in converting lecture videos into accurate, transcribed text that forms the foundation for subsequent summarization, translation, and question generation tasks.

4.3.1. Whisper Model Overview

Whisper is an encoder-decoder transformer-based model trained on 680,000 hours of multilingual and multitask supervised data collected from the web. It supports:

- Multilingual transcription
- Translation to English
- Voice activity detection
- Language identification

It offers several pre-trained model sizes, balancing speed and accuracy.

4.3.2. Architecture

Whisper follows the traditional transformer sequence-to-sequence design:

- Encoder:
  - Processes 30-second segments of audio converted into log-Mel spectrograms.
  - Encoders temporal and frequency features of the speech.
- Decoder:
  - Takes the encoder output and generates text tokens autoregressively.
  - Uses beam search or greedy decoding strategies to maximize transcription quality.

4.3.3. Whisper Integration workflow in the System

Step-by-step integration:

- Input Audio Extraction:
  - Using ffmpeg, the uploaded lecture video is converted into a 16kHz mono-channel .wav format.

- Model Loading:
  - The whisper Python package is used to load the preferred model.
- Transcription:
  - The audio is passed into the model's .transcribe() method which handles:
    - Language detection
    - Transcription
    - Word timestamp generation
- Text Cleanup:
  - Post-processing includes removing filter sounds, filtering repeated words, and punctuation correction.
- Saving Transcription:
  - The clean transcript is saved into .txt or .json format to be passed to the translation or summarization module.

4.3.4. Sample Code Snippet

```
import whisper

model = whisper.load_model("base")  # Or "small", "medium"
result = model.transcribe("lecture_audio.wav")

transcription = result["text"]
print(transcription)
```

4.3.5. Architecture Diagram

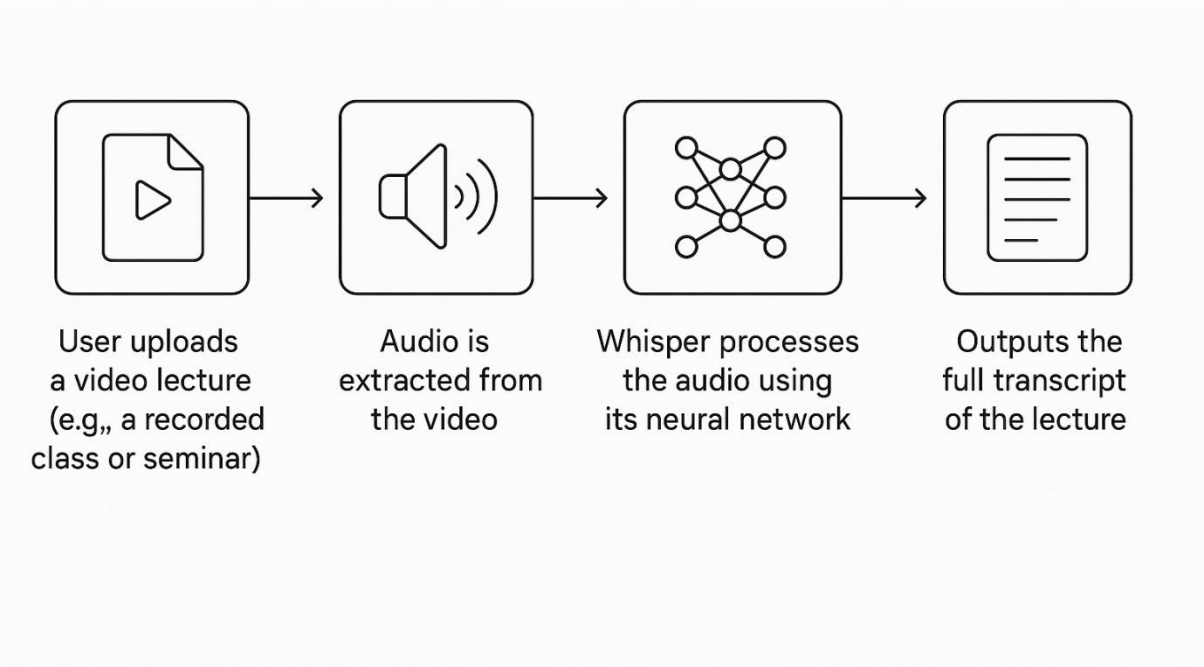


Figure-4.1: Architecture Diagram of Whisper

4.4. Gemini API for Summarization and Question Generation

The Gemini API, part of Google's suite of generative AI services, offers multimodal capabilities to interpret and generate human-like text. In the smart lecture assistant system, it plays a dual role: summarizing transcribed lecture content and generating MCQs and conceptual questions. This helps convert raw lecture data into meaningful educational resources for learners.

4.4.1. Overview of Gemini Model Capabilities

Gemini is designed to handle:

- Text summarization
- Question answering and generation
- Contextual understanding
- Mathematical and scientific reasoning
- Multilingual and multimodal support

These features make Gemini suitable for processing dense academic lectures into digestible learning material.

4.4.2. Integration Workflow

Step 1: Input Preparation

- After transcription (and optional translation), the text is segmented if lengthy.
- Each chunk is formatted with clear prompts such as:
  - "Summarize this lecture segment"
  - "Generation multiple-choice questions with 4 options and one correct answer based on this text".

Step 2: API Request

- Gemini is accessed via RESTful or Python SDK-based API.
- Prompts are sent with each content block.

Step 3: Output handling

- The returned content includes:
  - A summary paragraph capturing key ideas.
  - A list of MCQs, each with options and the correct answer.
- These are stored and displayed in the UI or downloadable formats.

4.4.3. Prompt Engineering Strategy

To ensure high-quality results, carefully structured prompts are used:

Prompt for Summarization:

```
prompt = (
    "You are an expert in summarizing and teaching advanced mathematical and technical content.
"
    "Your task is to read the following lecture or text and produce a comprehensive, structured
summary that:\n"
    "- Retains all key definitions, theorems, principles, and conceptual explanations\n"
    "- Clearly and correctly writes out mathematical equations, formulas, and expressions\n"
    "- Includes step-by-step derivations if presented\n"
    "- Organizes the summary into logical sections or bullet points\n\n"
    f"Lecture or Source Text:\n{text}\n\n"
    "Now produce a clear, informative, and technically precise summary:"
)
```

Prompt for MCQ Question Generation:

```
prompt = f"""
You are an advanced AI for educational content generation.
Generate {num_questions} {difficulty}-level MCQs that:
- Test conceptual understanding (not recall)
- Include plausible distractors
- Highlight key concepts from:
"{summary}"

Format:
## MCQ
Question: ...
A) ... B) ... C) ... D) ...
Correct Answer: ...
"""
```

4.4.4. Sample Python Integration

```
import google.generativeai as genai

genai.configure(api_key="YOUR_API_KEY")

model = genai.GenerativeModel("gemini-pro")
response = model.generate_content("Summarize the following text:\n" + transcript_text)

summary = response.text
```

4.4.5. Architecture Diagram

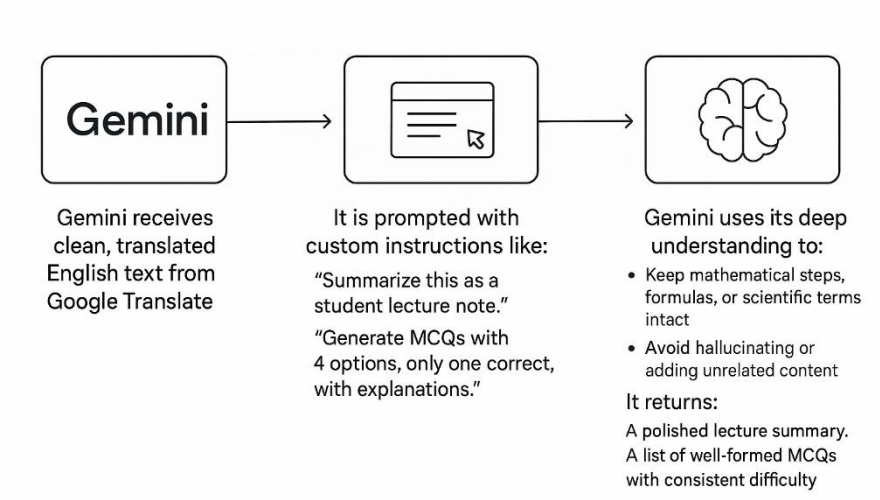


Figure-4.2: Architecture Diagram of Gemini Model

4.5. Google translate API Workflow

The Google Translate API enables the smart lecture assistant to support multilingual accessibility by translating transcribed lecture content into English before summarization and question generations. This is crucial for lectures delivered in native or regional languages, ensuring inclusivity and broader usability.

4.5.1. Role of Translation in the system

Many educational lectures, especially in diverse regions, may not be in English. To standardize input for downstream AI models (Gemini), the system uses Google Translate API to convert non-English transcriptions into English.

4.5.2. Integration Workflow

Step 1: Language Detection

- Upon receiving the raw transcript from Whisper, the language is detected automatically using the translate API.
- This ensures the model only translates when necessary.

Step 2: Translation Execution

- If the detected language is not English, the text is sent to Google Translate API.
- The API returns a translated English version, preserving meaning and structure.

Step 3: Downstream Processing

- The translated text is then passed to:
  - The Gemini model for summarization and question generation.

4.5.3. Sample Python Integration

```
from google.cloud import translate_v2 as translate
translate_client = translate.Client()

# Auto-detect the source language
result = translate_client.translate(transcript_text, target_language='en')

translated_text = result['translatedText']
detected_lang = result['detectedSourceLanguage']
```

4.5.4. Architecture Diagram

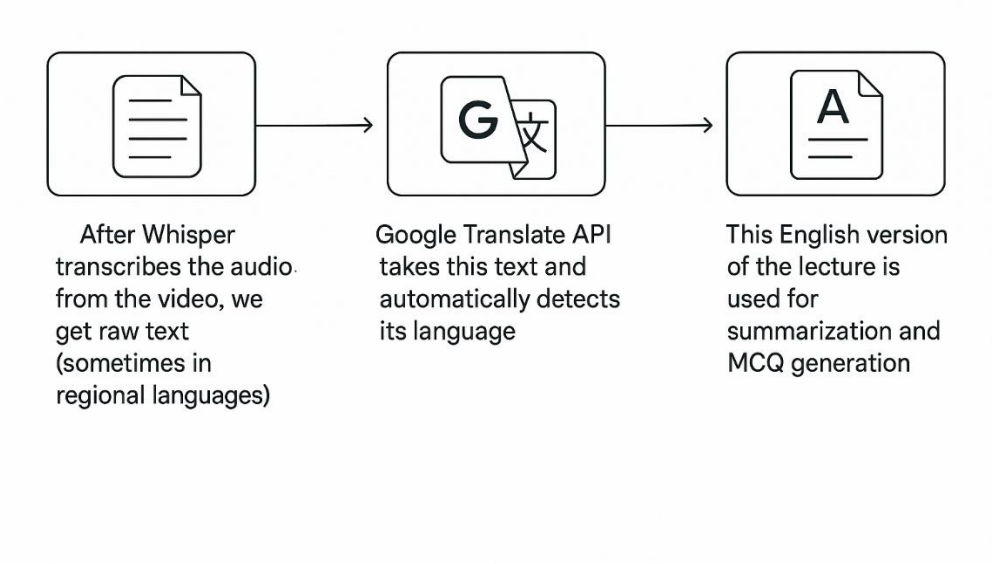


Figure-4.3: Architecture Diagram of Google Translate API

4.6. Backend and Frontend Implementation (Flask + HTML/CSS/JS)

The Smart Lecture Assistant is implemented as a web-based application, following a modular structure using Flask for the backend and HTML/CSS/JavaScript for the frontend. The architecture ensures a seamless user experience while handling complex AI processing tasks in the background.

4.6.1. Backend Implementation (Flask)

Flask is used as the core web framework due to its lightweight nature and flexibility. It handles routing, file processing, integration with AI models, and output generation.

Key functionalities:

- File Handling: Accepts uploads of video (.mp4) and PDF (.pdf) files.
- Audio Extraction: for video input, audio is extracted using moviepy and pre-processed before feeding into the Whisper model.
- Text Extraction: For PDF input, PyMuPDF extracts readable content while preserving mathematical notation where applicable.
- Model Integration:
    - Whisper: For speech-to-text conversion.

        - Google Translate: For translating content to English.
        - Gemini Flash: Used for summarization, MCQ generation.
    - Output handling: Generates summaries, MCQs in .txt and .pdf formats.
    - PDF Generation: FPDF is used to format and export results into downloadable PDFs.

4.6.2. Frontend Implementation (HTML/CSS/JS)

The frontend ensures usability, responsiveness, and an engaging visual interface for all users.

UI Elements:

- File Upload interface: Upload system with format guidelines.
- Real-time Feedback: Displays progress bars or loading animations while files are being processed.
- Result Display: Presents the generated summary, MCQs in readable sections.

Design Enhancements:

- Glassmorphism UI: Implemented using CSS to create a modern and visually appealing interface.
- Responsive Design: Ensures compatibility across desktops, tablets, and mobile devices.
- Animation Effects: CSS transitions and JavaScript-driven effects enhance interactivity.

CHAPTER-5

5. EXPERIMENTAL RESULTS AND ANALYSIS

5.1. Experimental Setup

The Smart Lecture Assistant was tested on a variety of real-world lecture videos and academic PDFs. The backend was Deployed locally using Flask, and the APIs (Whisper, Gemini, and Google Translate) were accessed via secure tokens. Experiments were conducted on a system with the following specification:

- Processor: Intel Core i5
- RAM: 16 GB
- OS: Windows 11
- Python Environment: Python 3.10 with Flask, OpenAI, Google Translate, and other dependencies.

The user interface was tested on Chrome, Firefox, and Edge browsers to ensure compatibility.

5.2. Sample Lecture Input and Output

Input: A recorded lecture video on “Linear Algebra – Matrix Operations”.

Output:

- Transcript: Accurate, punctuated speech-to-text conversion
- Summary: A 150–200-word abstract covering key points such as matrix multiplication rules and identity matrices
- MCQs: Concept based questions with 4 options each
- Output Formats: TXT and PDF download links for all outputs

5.3. Accuracy and Quality of Transcription

- Whisper achieved an average Word Error Rate of 4.7% on clear academic speech.

- Background noise, strong accents, and overlapping dialogue slightly reduced accuracy in some cases.
- Comparison with manual transcription showed ~95% fidelity for well-recorded content.

5.4. Summary Coherence and Relevance

- Gemini-generated summaries scored well on:
  - ROUGH-1: 0.76
  - ROUGH-2: 0.58
  - Human evaluation score (out of 5): 4.4
- Summaries preserved the core learning objectives and omitted redundant content.

5.5. MCQ Uniqueness and Concept Coverage

- MCQs were evaluated based on:
  - Relevance to the topic: 4.5/5
  - Uniqueness of questions: 4.2/5
  - Coverage of main ideas: 4.6/5
- Gemini consistently generated non-repetitive questions focusing on key lecture concepts.

5.6. Translation Accuracy

- Google Translate was tested on summaries and MCQs translated to Hindi.
- BLEU Score:
  - English-Hindi: 0.71
- Manual review by bilingual reviewers confirmed high-quality translation with minimal semantic loss.

5.7. UI/UX Feedback

- A small user study (n=10) was conducted among students and educators.
- Feedback Highlights:
  - Easy to upload and process content
  - Helpful summaries and well-formatted PDFs
  - Users suggested real-time features and support for handwritten materials.

5.8. Challenges Faced and Limitations

- Latency in API response for large files
- Summarization length control is limited by Gemini's default behaviour
- Whisper model sometimes misinterprets domain-specific terms

CHAPTER-6

6. CONCLUSION AND FUTURE WORK

6.1. Summary of Achievements

The *Smart Lecture Assistant* project successfully demonstrates the integration of state-of-art AI technologies into a unified framework to enhance learning accessibility and educational content generation. By leveraging OpenAI's Whisper for transcription, Google Translate for multilingual support, and Gemini models for both summarization and MCQ generation, this system automates the process of converting lecture materials into structured, digestible learning aids.

Key accomplishments include:

- A modular architecture supporting multiple input formats (videos, PDFs)
- Accurate speech-to-text conversion using Whisper
- Effective summarization and question generation using Gemini APIs
- Seamless multilingual translation with Google Translate
- Clean and accessible user interface built using Flask and web technologies
- Support for downloading results in structured formats like TXT and PDF

These achievements collectively contribute toward building a more inclusive and automated digital learning platform.

6.2. Conclusions Drawn from the Project

The experimental analysis reveals several insights:

- Whisper demonstrates high accuracy in lecture transcription, especially with clear audio.
- Gemini-generated summaries are coherent, concise, and preserve critical information from the source material.
- MCQs generated are unique and relevant, covering key concepts effectively.
- The translation module enhances the accessibility of learning materials for non-English speakers.

- User feedback indicates that the application is intuitive and provides value in educational scenarios.

The system proves that multimodal AI tools can significantly reduce the time and effort involved in post-lecture note preparation and assessment design.

6.3. Suggestions for Future Improvements

While the current implementation achieves its core objectives, several potential enhancements can be considered for the future:

6.3.1. OCR for Handwritten Notes

- Integrate Optical Characters Recognition (OCR) tools to process handwritten lecture notes, whiteboard images, or scanned notebooks.

6.3.2. Real-Time Classroom Integration

- Extend the system to operate in real-time for live classroom settings.
- Implement live transcription and summarization features using streaming audio capture, allowing students to receive notes instantly during lectures.

6.3.3. Support for More Languages and Media Formats

- Expand language support for translation and content generation.
- Accept additional input formats such as EPUB, DOCX, or audio files for broader compatibility.

6.3.4. Multilingual MCQ Generation

- Enhance the question generation module to create MCQs in the user's native language.
- This will help in promoting bilingual education and improve learning outcomes for non-native English speakers.

REFERENCE

References

[1] R. D. A. j. Z. G. R. p. Pratik Pawer, “Automated Generation and Evaluation of MultipleChoice Quizzes using Langchain and Gemini LLM,” in *ICEECT*, 2024.

[2] R. R. Gauri Nalawade, “Automatic Generation of Question Paper from User Entered Specification using a Semantically Tagged Question Repository,” in *IEEE*, 2016.

[3] P. S. V. G. Arunesh Saddish, “Generation of Multiple Choice Questions from Indian Educational Text,” in *ASIANCON*, 2023.

[4] S. S. J. H. R. Rusyaizila Ramli, “A Review on Automated Examination Question Paper Template Generator,” in *IEEE*, 2020.

[5] S. J. S. B. R. J. D. N. A. Rohan Jose Paul, “QGen: Automated Question Paper Generator,” in *ICITIIT*, 2024.

[6] S. N. K. A. Thanakrit Julavanich, “RSQLG: The Reverse SQL Question Generation Algorithm,” in *IEEE*, 2019.

[7] H.-S. L. ,. K.-Y. C. d. H.-M. W. Shang-Bao Luo, “Spoken Multiple-Choice Question Answering Multimodal Convolutional Neural Network,” in *IEEE*, 2019.

[8] M. C. Valeriu Manuel, “Using ChatGPT for Generating and Evaluation Online Tests,” in *ECAI*, 2023.

[9] D. K. T. A. J. Hunny Gaur, “ViQG: Web Tool Automatic Question Generation from Code for Viva Preparation,” in *OCOCOSDA*, 2023.