



SCHOOL OF INFORMATION AND COMMUNICATION TECHNOLOGY GAUTAM
BUDDHA UNIVERSITY, GREATER NOIDA, 201312, U. P., (INDIA)

Candidate's Declaration

We hereby certify that the work embodied in this **Internship Report**, submitted in partial fulfilment of the requirements for the award of the degree of M.Sc. Computer Science, to the School of Information and Communication Technology, Gautam Buddha University, Greater Noida, is an authentic record of our own work carried out during the internship. This report is based on the learning and experience gained under the supervision of **Prof. Prakash Kumar Saraswat**, School of ICT. The content presented in this report is original and has not been submitted to any other University or Institute for the award of any other degree or diploma.

NAME	ROLL NO	SIGNATURE
NISHANT GOEL	235/PMD/001	

Supervisor's Certification

This is to certify that the above statement made by the candidates is correct to the best of my knowledge and belief. However, responsibility for any plagiarism-related issue solely stands with the students.

Signature of the Supervisor:

Name with Designation: **Prof. Prakash Kumar Saraswat** (Supervisor)

Date:

Place: Greater Noida

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to all those who supported and guided me throughout the course of my internship project.

First and foremost, I extend my heartfelt thanks to **Prof. Prakash Kumar Saraswat**, whose constant encouragement, valuable insights, and expert guidance have been instrumental in shaping this internship experience into a meaningful learning journey.

I am also profoundly grateful to Mr. Saurabh Gupta, my Team Manager at Unicloud, for his continuous support, mentorship, and for providing me with the opportunity to be a part of challenging real-world projects. His constructive feedback and technical guidance have greatly enriched my practical knowledge and skill set.

I would also like to thank all the team members and staff at Unicloud for their cooperation and assistance, which made my internship a productive and enjoyable experience.

Lastly, I acknowledge the support of my institution, Gautam Buddha University, for facilitating this internship program and helping me take this significant step in my professional development.

List Of Tables

1. Chapter-1(Introduction)	6
1.1. Background Of The Project Chapter-1(Introduction)	6
1.2. Objective	7
1.3. Purpose, Scope and Availability	7
2. Chapter-2(Details Of Internship)	8
2.1. Internship Experience at UniCloud Technologies Pvt. Ltd	8
2.2. Existing System	9
2.3. Limitation of Exiting System	9
2.4. Proposed System	10
2.5. Benefit Of Proposed System	11
2.6. Feature of Proposed System	11
2.7. System Requirements	12
3. Chapter-3(System Design)	14
4. Chapter-4(ER Diagram)	16
5. Chapter-5(Algorithm)	18
5.1. Intellipdf	18
5.2. Automated Question Paper Generation	19
6. Conclusion	21
7. Chapter-7(References)	22

List Of Figures

1. Fig.3.1	14
2. Fig.3.2	15
3. Fig.4.1	16
4. Fig.4.2	17
5. Fig.5.1	19
6. Fig.5.2	20

ABSTRACT

In the era of digital transformation, organizations and individuals increasingly rely on large volumes of unstructured data such as PDF documents, multimedia content, and real-time visual feeds. Extracting meaningful insights from such diverse formats presents a significant challenge, especially when speed, accuracy, and automation are essential. This project addresses these challenges by proposing a set of intelligent AI-powered tools for natural language interaction, information retrieval, and structured data generation from both text and video sources.

The major highlight of this project is a system called “IntelliPDF: AI-Powered Document Interaction and Real-Time Vision Analytics,” which allows users to interact with static PDF documents using natural language queries. By leveraging advanced models like Google’s Gemini 1.5 Pro and various open-source LLMs such as Mixtral-8x7B, Zephyr-7B, and Baichuan-13B, the system can accurately retrieve and respond to user queries from large document sets. In addition, another key module focuses on educational automation—extracting questions from academic PDFs and converting them into a structured Excel-based format, ideal for building question banks or mock papers

.

Complementary contributions include a dialogue summarization system using a PEFT-tuned Phi-2 model with LoRA adapters, a real-time fire detection system using YOLOv8, and a people detection interface for bounding box visualization in video feeds. The project also features a portfolio-style internship website to showcase the complete integration and usability of all tools. Collectively, these systems automate document understanding, enhance multimedia analysis, and demonstrate real-world application of AI in academic and industrial contexts.

CHAPTER - 1

INTRODUCTION

This chapter introduces the context of the project, defines its objectives and scope, and outlines the purpose and structure of the report.

1.1 BACKGROUND OF THE PROJECT

In the age of digital transformation, the volume of documents generated, stored, and processed has grown exponentially. Extracting relevant information from these documents, particularly in PDF format, is time-consuming and challenging without intelligent tools. Traditional keyword-based search methods lack contextual understanding, making them inefficient for detailed document exploration. To address this issue, the project titled “**IntelliPDF: AI-Powered Document Interaction and Real-Time Vision Analytics**” was developed to facilitate natural language interaction with PDF files using state-of-the-art AI and NLP technologies.

With the increasing demand for automated educational tools and intelligent search systems, the need for a robust solution that can summarize content, answer user queries, and extract structured information has become critical. In addition to the main system, the project integrates several complementary modules. These include a **question extraction tool** that converts academic PDFs into structured Excel formats, **real-time fire and people detection** using YOLOv8, and a **demo portfolio website** built as part of an internship experience. Another notable module involves **dialogue summarization using a LoRA-adapted Phi-2 model**, where PEFT techniques are applied to efficiently generate concise summaries from long conversations. This contributes to tasks like meeting summarization, customer service automation, and educational content processing.

These innovations collectively demonstrate the use of cutting-edge models such as Google Gemini Pro 1.5, Mixtral-8x7B, Zephyr-7B, Baichuan-13B, and Microsoft Phi-2 with LoRA. Together, they enable intelligent querying, automated summarization, and content generation across multiple formats. The complete solution reflects strong applicability in both academic and industrial domains, while significantly reducing manual effort and enhancing document interaction.

1.2 Objectives

This project aims to:

- Develop an AI-powered system for querying PDF documents using natural language, removing the need for manual search.
- Build an interactive Streamlit chatbot for real-time document interaction.
- Extract structured academic content (e.g., MCQs and descriptive questions) from PDFs and export to Excel.
- Implement real-time object detection for people and fire using YOLOv8 to support safety applications.
- Design a portfolio website to present internship modules and contributions.
- Evaluate and integrate multiple LLMs (Gemini Pro, Mixtral, Zephyr, Baichuan, MiniLM) for embedding and response generation.
- Use Phi-2 with LoRA adapters for efficient dialogue summarization.

1.3 Purpose, Scope, and Applicability (Short Version)

Purpose:

The project addresses the challenge of extracting insights from unstructured PDFs by enabling natural language querying and delivering context-aware answers. It combines state-of-the-art models like Gemini Pro, Mixtral, Baichuan, and Phi-2 with modules for question generation, summarization, and real-time vision analytics, offering a powerful alternative to tools like ChatPDF.

Scope:

Designed for academic and research users, the system supports offline and customizable document analysis. It leverages text extraction, embeddings (MiniLM/Gecko), FAISS retrieval, and LLMs for accurate responses. Video analysis via YOLOv8 enhances applicability in safety and surveillance contexts.

Applicability:

Usable across education, research, and public safety, the tool enables fast info retrieval, automated question creation, and summarization. With offline support and local deployment, it stands out as a scalable and accessible AI system for real-world use.

CHAPTER -2

Details of the Internship

2.1 Internship Experience at UniCloud Technologies Pvt. Ltd.

As part of my MSc program, I completed a 3-month internship at UniCloud Technologies, a company specializing in cloud infrastructure and AI solutions. The internship offered hands-on experience in machine learning, NLP, and deploying models in real-world environments.

Company Overview:

- Cloud Infrastructure & Virtualization
- AI/ML Solutions (NLP, Computer Vision)
- DevOps & CI/CD Automation
- SaaS Platform Development

Internship Highlights:

- Model Fine-Tuning: Worked on LLaMA 3 8B, Mistral 7B, and Gemma 2B using LoRA and 4-bit quantization for efficient deployment.
- Inference Pipelines: Built scalable pipelines for dialogue summarization with LangChain and Streamlit.
- DevOps & Deployment: Gained experience in containerization, cloud integration, and deploying production-ready AI models.

Skills Gained:

- Transformer models, NLP, and computer vision
- Tools like Hugging Face, Google Colab, Kaggle, and VS Code
- Knowledge of LoRA fine-tuning, quantization, and cloud-based deployment

Conclusion:

This internship significantly boosted my technical and professional skills, providing real-world exposure to advanced AI/ML systems and industry practices.

2.2 Existing System

Several AI platforms like ChatPDF, AskYourPDF, and Humata enable users to query PDF documents in natural language. These tools break content into chunks, embed them using models (e.g., OpenAI embeddings, sentence-transformers), and retrieve relevant information. While effective, they have notable limitations:

- **Usage Restrictions:** Free plans often limit file size, query count, and document uploads.
- **Privacy Concerns:** As cloud-based systems, they risk exposure of sensitive data.
- **Lack of Customization:** These platforms don't allow integration of custom models, embedding controls, or UI modifications, and are not open-source—hindering offline use and institutional control.

In object detection, models like YOLOv5, YOLOv7, and YOLOv8 excel in tasks such as people or fire detection. However, they are typically standalone modules, not integrated with document processing systems, making multi-purpose deployments inefficient.

For question generation, many educational tools rely on manual or semi-automated methods, with limited automation in converting academic content into structured formats like Excel. Few systems offer flexible formatting or tagging.

In text summarization, advanced models like BART, T5, GPT-4, and Pegasus perform well but require heavy hardware and are often not fine-tuned for domain-specific tasks (e.g., dialogue summarization). Lightweight PEFT methods like LoRA are rarely used with accessible models (e.g., Microsoft Phi-2) in practical settings.

Overall, while strong individual tools exist for document querying, object detection, question generation, and summarization, they remain fragmented and lack integration. No current solution offers a unified, customizable, and locally deployable framework combining these capabilities. This project addresses that gap by providing an all-in-one, accessible, and flexible system.

2.3 Limitations of the Existing System

Current AI tools like ChatPDF and Humata have limits on queries, file size, and uploads, often requiring paid plans. They process documents on cloud servers, raising privacy

concerns. These closed-source platforms lack customization, multi-model support, and offline use.

Most cannot automatically extract questions from PDFs or export them to Excel for easy question bank creation. Object detection models like YOLOv8 aren't integrated with document tools, missing unified workflows. Summarization tools like GPT-4 are expensive and hardware-heavy, with limited use of affordable tuning methods like LoRA.

Summary:

Existing systems are restrictive, not flexible, and poorly integrated. This project aims to build a unified, customizable, and offline-capable solution.

2.4 Proposed System

In today's data-driven world, extracting meaningful information from PDFs and videos is challenging. Traditional keyword searches are inefficient, and commercial tools like ChatPDF offer limited customization, lack transparency, and don't support offline or academic needs. They also miss real-time safety monitoring features.

This project proposes an integrated, intelligent system combining document querying, academic content extraction, dialogue summarization, and real-time object detection in one modular, locally hostable platform with a user-friendly web interface.

Key Modules:

1. **PDF Chatbot:** Uses embeddings (Gecko, MiniLM) stored in FAISS and LLMs (Gemini Pro, Mixtral-8x7B, Zephyr-7B) to answer user queries contextually.
2. **Academic Question Extraction:** Automatically pulls questions from PDFs and exports them to Excel, aiding educators.
3. **Dialogue Summarization:** Phi-2 model fine-tuned with LoRA efficiently summarizes long conversations on modest hardware.
4. **Real-Time Detection:** YOLOv8 monitors video feeds for fire and people detection, improving safety surveillance.
5. **Unified Interface:** Streamlit-based UI for easy uploads, AI interaction, video monitoring, and output downloads.

2.5 BENEFITS OF THE PROPOSED SYSTEM

The proposed system offers a unified, open-source solution that enhances document querying, educational automation, dialogue summarization, and real-time object detection. Its key benefit is enabling natural language interaction with PDF documents, making it faster and easier to extract information. It supports multiple advanced language models (Gemini Pro, Mixtral-8x7B, Zephyr-7B, Baichuan-13B), offering flexibility and scalability across cloud or local setups. Unlike commercial tools, it's customizable, offline-capable, and free of usage limits. The question extraction module automates conversion of questions into structured Excel files, saving educators time. Dialogue summarization is optimized with lightweight LoRA-based Phi-2 models, allowing efficient summarization without high-end hardware. The system also integrates YOLOv8 for fire and people detection, broadening its scope to surveillance and safety. A Streamlit web interface makes it easy to use for all, promoting research and innovation in AI and Computer Vision.

2.6 FEATURES OF THE PROPOSED SYSTEM

The system combines document intelligence, academic automation, conversational summarization, and real-time video analysis. Key features include:

- **Natural Language PDF Querying:** Chat-based interface powered by vector search and large language models.
- **Multi-Model Support:** Works with Gemini Pro, Mixtral, Baichuan, Zephyr, and multiple embedding models.
- **Question Extraction:** Detects questions from PDFs and exports them to Excel in a structured format.
- **Dialogue Summarization:** Uses Phi-2 with LoRA for efficient, domain-specific summarization.
- **Object Detection:** YOLOv8 detects fire and people in real-time video streams.
- **Streamlit Web App:** User-friendly, browser-based interface with no technical setup needed.
- **Local Deployment:** Fully offline-capable, no subscriptions, customizable, and open source.

2.7 SYSTEM REQUIREMENTS SPECIFICATION

System Must:

- Support natural language queries on PDFs with accurate context-aware answers.
- Extract questions from academic PDFs and export them to Excel.
- Summarize long dialogues using LoRA-tuned Phi-2 models efficiently.
- Detect fire and people in videos in real time with customizable settings.
- Integrate all features into a web app showing uploads, queries, summaries, and video analysis.
- Provide smooth performance, intuitive UI, offline capability, and flexibility for advanced users.

User Characteristics

- **Students:** Query PDFs, summarize transcripts.
- **Teachers:** Automate question bank creation.
- **Researchers:** Extract and compare model outputs.
- **Content Creators:** Build quizzes and learning materials.
- **Developers:** Customize and extend the system.
- **Admins:** Generate summaries and reports.
- **Security Staff:** Monitor videos for safety.
- **Institutions:** Manage access with roles.

Software and Hardware Requirements by Module

1. IntelliPDF (Document Interaction & Vision Analytics):

- OS: Windows 10/11, Ubuntu 20.04+, macOS
- Python 3.10+, libraries like streamlit, PyPDF2, langchain, sentence-transformers, faiss-cpu, torch

- CPU: Intel i5/Ryzen 5; RAM: 8–16 GB; Storage: 10–20 GB; GPU optional

2. Question Extraction & Excel Generation:

- Python 3.10+, PyPDF2, pandas, openpyxl, streamlit, langchain, transformers
- CPU: Dual-core; RAM: 4–8 GB; Storage: ~5 GB; GPU not required

3. Dialogue Summarization (Phi-2 + LoRA):

- Python 3.10+, datasets, transformers, peft, torch
- CPU: Intel i5; RAM: 8–16 GB; Storage: 5–10 GB; GPU optional (4GB+ recommended)

4. Real-Time Detection (YOLOv8):

- Python 3.10+, ultralytics, opencv-python, torch, numpy, streamlit or custom UI
- CPU: Intel i7/Ryzen 7; RAM: 8–16 GB; Storage: 10–15 GB; GPU strongly recommended (NVIDIA GTX 1650+)

5. Web App Integration & Demo:

- Frontend: HTML, CSS, JS (React optional)
- Backend: Streamlit/Flask, with all module dependencies
- CPU: Dual-core+; RAM: 8 GB; Storage: ~10 GB; GPU not mandatory

Summary:

This modular breakdown ensures each component of the system has the right tools and platform to function effectively. While some modules are lightweight and work on general-purpose laptops, others — like YOLOv8 detection — benefit greatly from GPU acceleration. By designing flexible hardware/software requirements for each module, the system can be scaled and deployed in both **resource-constrained academic labs** and **production-grade environments**.

CHAPTER -3

System Design

The system design phase transforms the problem requirements into a blueprint for construction. It includes architectural design, data flow, relationships among entities, user interface layout, and output report structures. The system is modular, with each module addressing a specific aspect of document processing, AI-based summarization, detection, or web deployment.

Module 1: IntelliPDF: AI-Powered Document Interaction and Real-Time Vision Analytics (AIPowered PDF Question Answering)

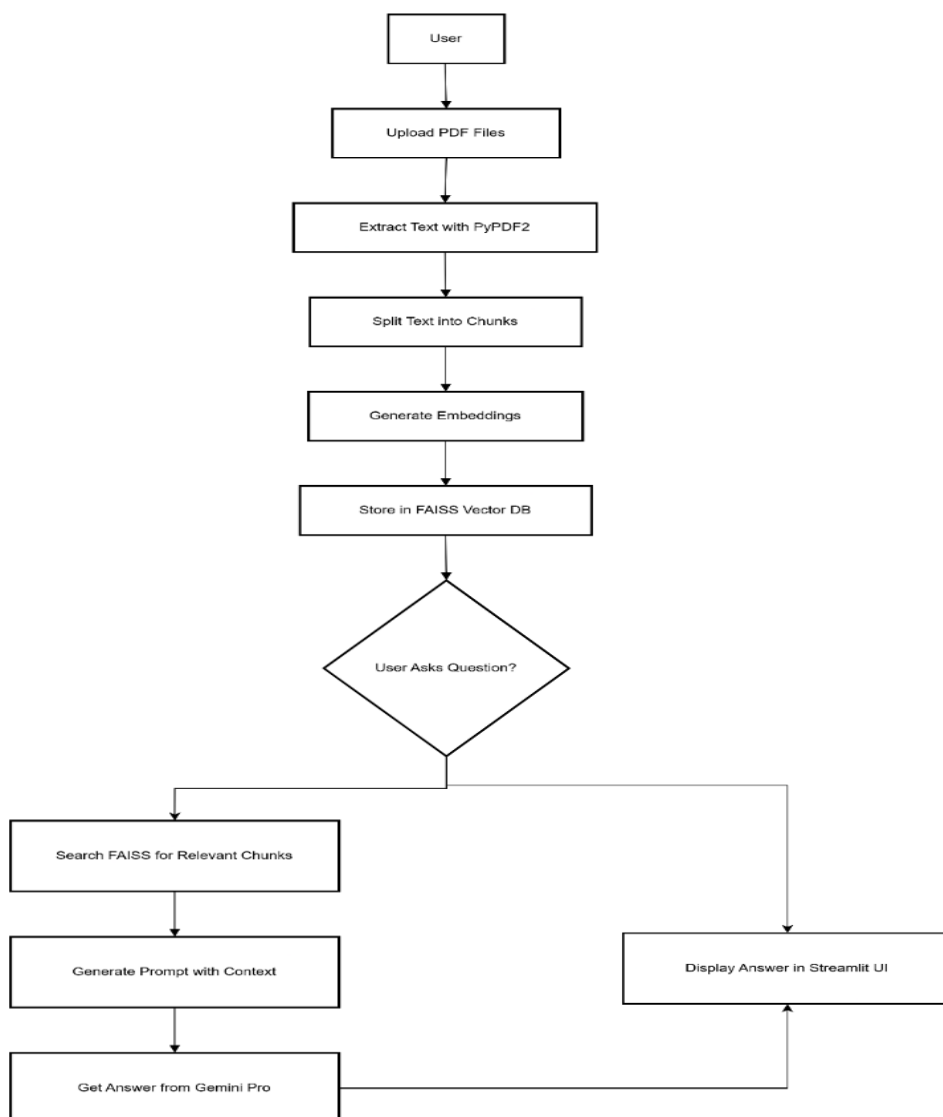


Fig.3.1

Module 2: Question Generator from PDF

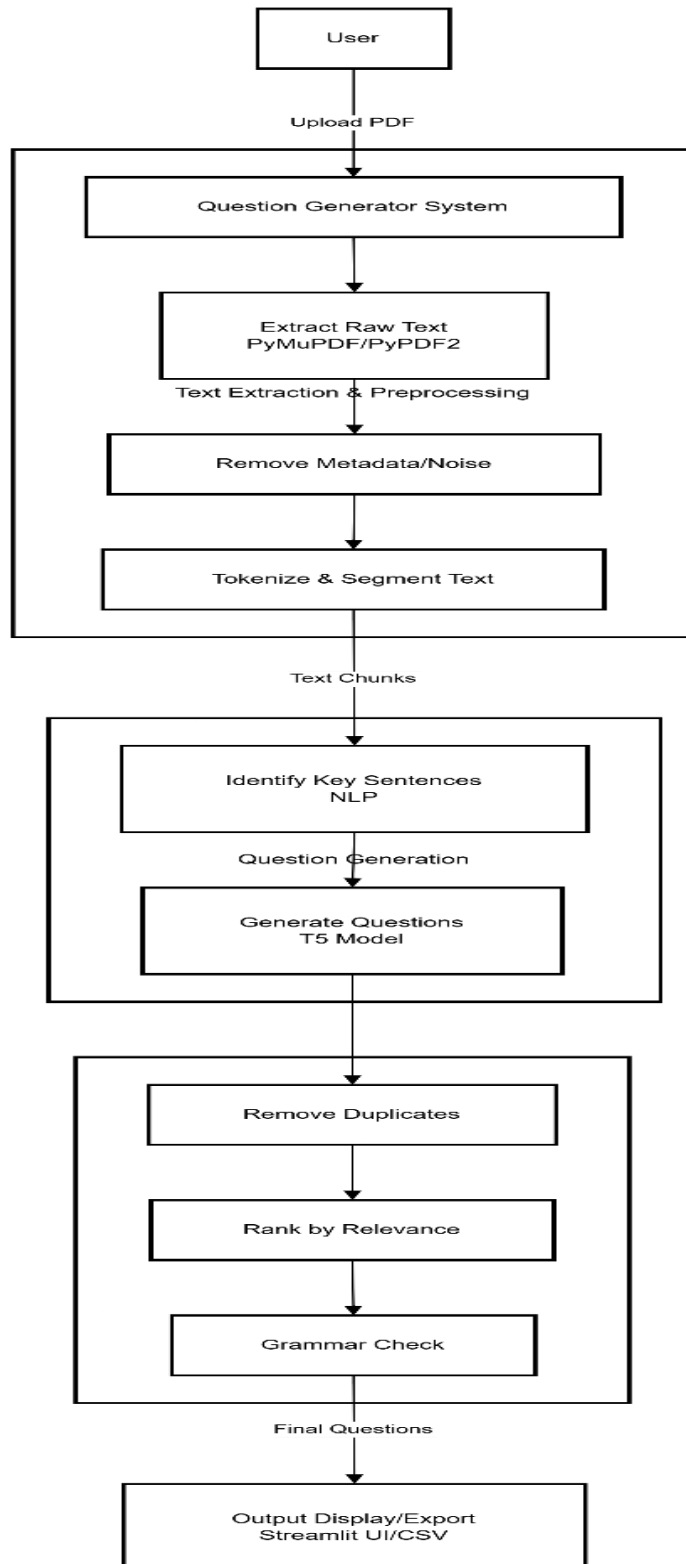


Fig.3.2

CHAPTER -4

ER DIAGRAM

This section outlines the relational structure of the system. The ER diagram represents the core entities involved in the **IntelliPDF: AI-Powered Document Interaction and Real-Time Vision Analytics** system, the relationships between them, and how data flows from document upload to user query resolution.

Module 1: IntelliPDF: AI-Powered Document Interaction and Real-Time Vision Analytics – Entity Relationship Diagram

The entity-relationship model for Module 1 captures the flow of user-uploaded PDFs through the document processing pipeline into searchable chunks and embeddings. It also maps how users interact with the system through queries and receive intelligent responses.

Entity Relationships Overview

- **USER** uploads multiple **PDFs**
- Each **PDF** contains many **TEXT_CHUNKS**
- Each **TEXT_CHUNK** has one corresponding **EMBEDDING**
- A **USER** can make many **QUERY** entries
- Each **QUERY** generates a **RESPONSE**

This logical structure supports scalable document analysis, search, and conversational interfaces via AI.

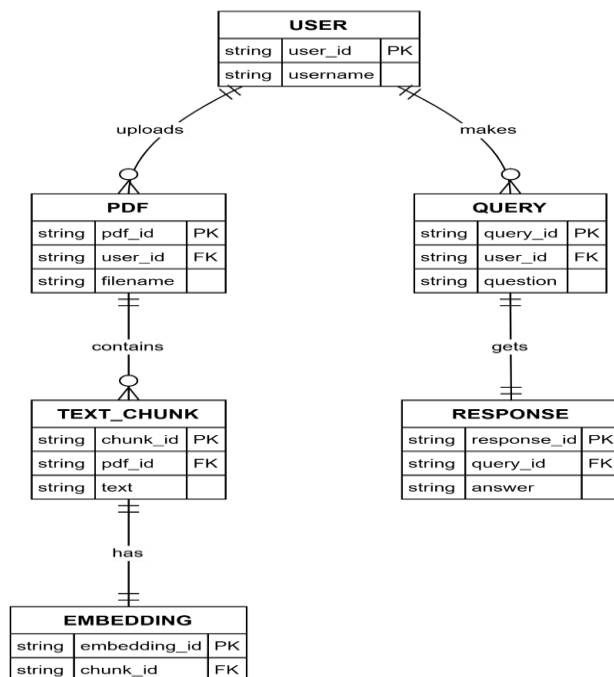


Fig.4.1

Module 2: Question Generator from PDF – Workflow Explanation

The **Question Generator from PDF** module is designed to automatically extract, segment, and transform educational PDF content into meaningful questions. This system supports both student and educator roles and enables export of questions in multiple formats such as CSV or UI-based test interfaces. The ER diagram highlights the flow of data and the interrelationship between key components.

Workflow Based on ER Diagram

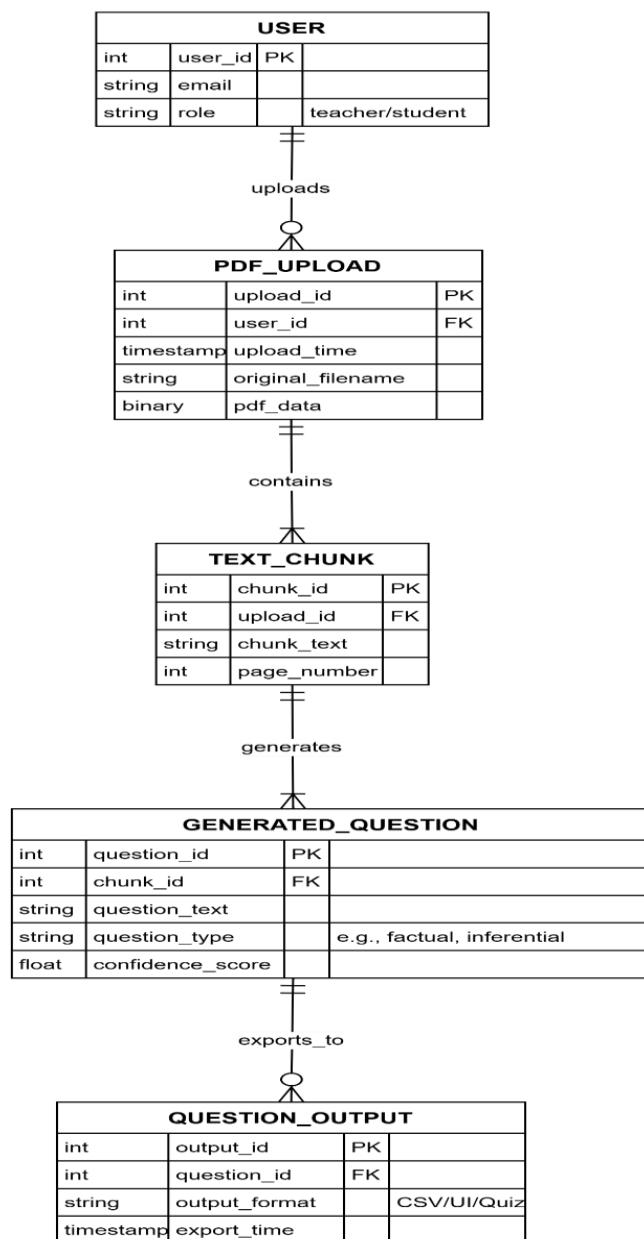


Fig.4.2

CHAPTER -5

ALGORITHM

5.1 Algorithm: IntelliPDF: AI-Powered Document Interaction and Real-Time Vision Analytics Using LangChain and Gemini Pro

ModuleName: Natural Language PDF Question Answering

Objective: Enable semantic interaction with PDF documents using a conversational AI interface

Input: One or more PDF files uploaded by the user; a user-typed question

Output: A relevant, well-formed answer extracted and generated from document content

Step-by-Step Algorithm Description

Step 1: Load API Key Securely

- Use dotenv to load GOOGLE_API_KEY.
- Configure Gemini API to avoid hardcoding credentials.

Step 2: Upload & Extract PDF Text

- Use st.file_uploader to upload PDFs.
- Extract text from each page using PyPDF2.
- Combine all text into one string for processing.

Step 3: Chunk & Preprocess Text

- Split text into 1000-character chunks (200-character overlap) using RecursiveCharacterTextSplitter from LangChain.

Step 4: Generate Embeddings & Index with FAISS

- Use GoogleGenerativeAIEmbeddings to convert chunks to vectors.
- Store vectors in FAISS for fast semantic search.
- Save the index to disk.

Step 5: User Query → Semantic Search

- User types question via st.text_input.
- Convert query to vector and find top relevant chunks using FAISS.

Step 6: Generate Answer via Gemini Pro

- Use gemini-1.5-pro with a custom prompt.
- Run `load_qa_chain()` with stuff mode to combine context and question.
- Model answers only from retrieved chunks or says "not available in context."

Step 7: Display Results & Handle Errors

- Show the final answer in Streamlit.
- Handle exceptions, show progress with spinners, and allow re-generation or export.

Output

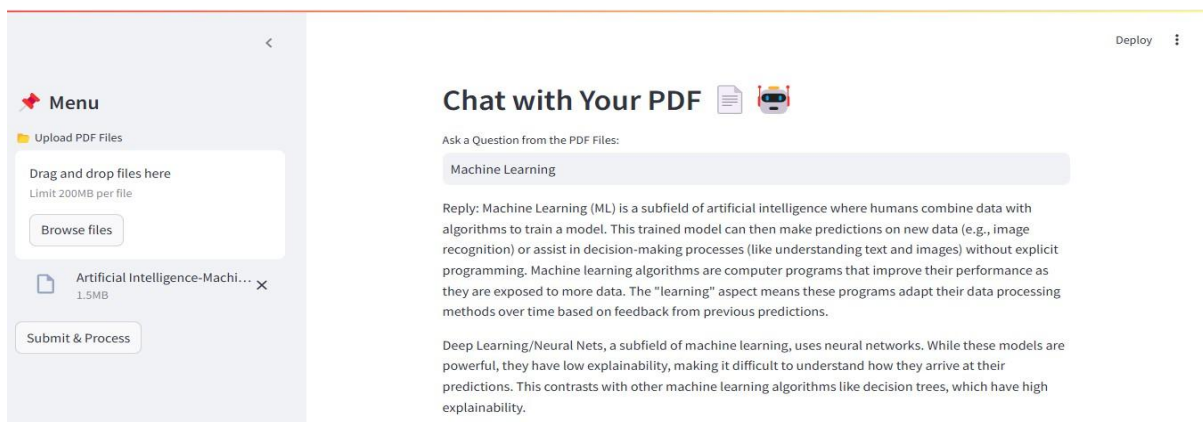


Fig.5.1

5.2 Automated Question Paper Generator System

Step 1: Document Ingestion

- Upload PDFs via web or API.
- Validate file type/size.
- Extract text using PyPDF2 (with fallback loader).
- Clean text (remove headers, footers, special chars, fix spacing).

Step 2: Content Processing

- Split text into ~1500-char chunks with 300-char overlap.
- Normalize (lowercase, clean symbols).
- Preserve structure: headings, bullet points, lists.

Step 3: Question Generation

- Use a local LLM (e.g., Mistral-7B) with:
 - temperature=0.5, top_p=0.9, max_tokens=1024
- Generate:
 - MCQs with 4 options, correct answer, (optional explanation)
 - Descriptive questions (50–100 words)
- Add retry logic (up to 3 tries, context trimming, batching).

Step 4: Paper Assembly

- Validate pool against config (counts, types, marks).
- Organize into sections (MCQ, Descriptive).
- Rank by quality/confidence, avoid duplicates.
- Handle shortages with fallback logic or notify user.

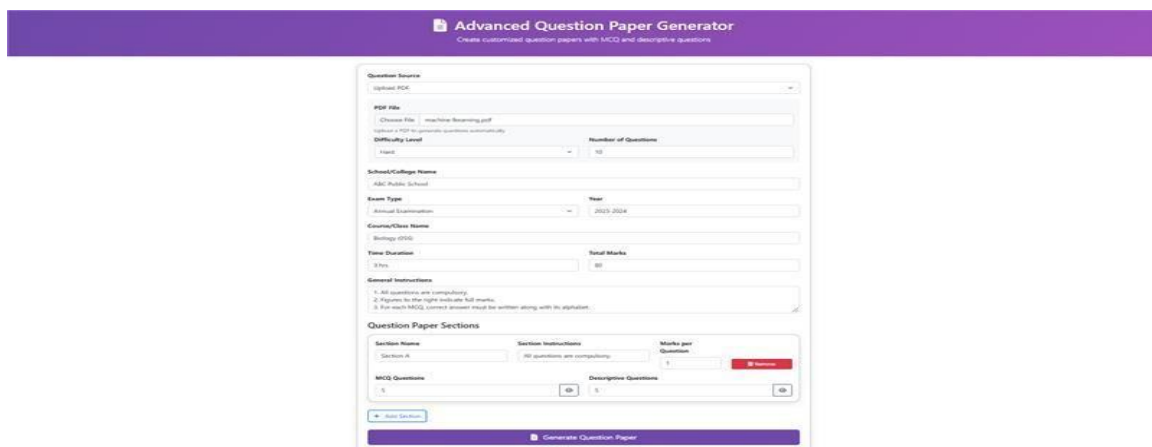
Step 5: Output Generation

- Export to:
 - Styled PDF (with pagination, headers, marks)
 - CSV (with metadata)
 - HTML preview (responsive UI with section markers)

Step 6: Delivery System

- Provide REST API endpoints.
- Real-time feedback with spinners, errors, and confirmations

Output



The screenshot displays the 'Advanced Question Paper Generator' web application. The interface is divided into several sections for configuring a question paper. At the top, there's a header with the title and a subtitle 'Create customized question papers with MCQ and descriptive questions'. Below this, the 'Question Source' section allows users to 'upload PDF' or 'choose file' (with a note that a PDF is generated automatically). The 'Difficulty Level' is set to 'Hard' and the 'Number of Questions' is '10'. The 'School/College Name' is 'ABC Public School'. The 'Exam Type' is 'Annual Examination' and the 'Year' is '2023-2024'. The 'Course/Class Name' is 'Biology (10th)'. The 'Time Duration' is '3 hrs' and the 'Total Marks' is '80'. The 'General Instructions' section lists three rules: 1. All questions are compulsory, 2. Answers to the right indicate full marks, and 3. For each MCQ, correct answer must be written along with its alphabet. The 'Question Paper Sections' table shows 'Section A' with '10 questions are compulsory' and 'Marks per Question' of '1'. Below this, there are input fields for 'MCQ Questions' (set to 5) and 'Descriptive Questions' (set to 5). A 'Generate Question Paper' button is at the bottom right.

Fig.5.2

CHAPTER -6

CONCLUSION

Key Takeaways

- Learned to integrate AI, NLP, and web technologies for building intelligent systems.
- Gained hands-on experience in LLMs, text vectorization (MiniLM, FAISS), and querying PDFs using models like Gemini Pro, Mixtral, Zephyr.
- Applied LoRA-based fine-tuning on Phi-2 for dialogue summarization.
- Built interactive frontends using HTML, JS, and Streamlit, and designed system diagrams using draw.io.
- Developed complete pipelines: text extraction, embedding, model inference, and UI integration.

Key Contributions

- IntelliPDF: AI-based document query system using Gemini & FAISS.
- Question Generator: Extracts and generates exam questions from PDFs.
- Dialogue Summarizer: Phi-2 + LoRA model for summarizing conversations.

Limitations

- Accuracy drops for scanned/multilingual documents.
- Hardware dependency for real-time inference.
- UI lacks feedback loop for refining generated summaries.

Future Scope

- Add OCR, multilingual support, and interactive editing.
- Improve question filtering and tagging.
- Integrate summarization into real-world systems like CRMs.

CHAPTER-7

REFERENCES

- [1] P.E. Agre and D. Chapman, *Unpublished memo*, MIT Artificial Intelligence Laboratory, Cambridge, MA, 1986.
- [2] R.J. Bobrow and J.S. Brown, “Systematic understanding: synthesis, analysis, and contingent knowledge in specialized understanding systems,” in *Representation and Understanding*, R.J. Bobrow and A.M. Collins, Eds. New York: Academic Press, 1975, pp. 103–129.
- [3] R.A. Brooks, “A robust layered control system for a mobile robot,” *IEEE J. Robotics and Automation*, vol. 2, no. 1, pp. 14–23, 1986.
- [4] R.A. Brooks, “A hardware retargetable distributed layered architecture for mobile robot control,” in *Proc. IEEE Int. Conf. Robotics and Automation*, Raleigh, NC, 1987, pp. 106–110.
- [5] R.A. Brooks, “Intelligence without representation,” *Artificial Intelligence*, vol. 47, no. 1–3, pp. 139–159, 1991.
- [6] S. Thrun, “Learning metric-topological maps for indoor mobile robot navigation,” *Artificial Intelligence*, vol. 99, no. 1, pp. 21–71, 1998.
- [7] L. Steels, “Cooperation between distributed agents through self-organization,” in *Decentralized AI*, Y. Demazeau and J.P. Müller, Eds. Amsterdam: North-Holland, 1990, pp. 175–196.
- [8] R.C. Arkin, *Behavior-Based Robotics*. Cambridge, MA: MIT Press, 1998.
- [9] M.J. Mataric, “Behavior-based robotics as a tool for synthesis of artificial behavior and analysis of natural behavior,” *Trends in Cognitive Sciences*, vol. 2, no. 3, pp. 82–87, 1998.
- [10] L.P. Kaelbling, M.L. Littman, and A.W. Moore, “Reinforcement learning: A survey,” *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996.
- [11] Google AI, *Google Generative AI – Gemini API*, accessed March 2025. [Online]. Available: <https://ai.google.dev/>
- [12] Hugging Face, *HuggingFace Model Hub – Transformers Library*, accessed February 2025. [Online]. Available: <https://huggingface.co/docs/transformers>
- [13] Unsloth Team, *Unsloth: Efficient LoRA and PEFT for LLMs*, accessed February 2025. [Online]. Available: <https://unsloth.ai>
- [14] PyPDF2 Library, *PDF Extraction in Python*, accessed January 2025. [Online]. Available: <https://pypi.org/project/PyPDF2/>

- [15] Facebook AI Research, *FAISS: A Library for Efficient Similarity Search*, accessed January 2025. [Online]. Available: <https://github.com/facebookresearch/faiss>
- [16] OpenCV, *Open Source Computer Vision Library*, accessed January 2025. [Online]. Available: <https://opencv.org/>
- [17] Ultralytics, *YOLOv8: Real-Time Object Detection Models*, accessed February 2025. [Online]. Available: <https://docs.ultralytics.com/>
- [18] Roboflow, *Labeling and Training YOLOv8 Models*, accessed January 2025. [Online]. Available: <https://roboflow.com/>
- [19] LangChain Documentation. *LangChain: Building Applications with LLMs*. Accessed March 2025. <https://docs.langchain.com>
- [20] Microsoft. *Phi-2 Model*. Hugging Face, Accessed March 2025. <https://huggingface.co/microsoft/phi-2>