# A Unified Privacy-Preserving AI Assistant Integrating Conversational AI, Legal Advisory, and Document Intelligence

[1]Arpita Dhanpal Devmore, [2]Bhagyashree Sanjay Deshmukhe, [3]Manasi Mahavir Kurade, [4]Girish Rahul Bhosale, [5]Prasanna Sanjay Adake, [6]Mr. Shrenik .R .Patil (Guide)

[1,2,3,4,5] UG Students, [6] Assistant Professor , *Corresponding Author
Department of Computer Science & Engineering
D.K.T.E. Society's Textile & Engineering Institute, Ichalkaranji, India

**DOI**: 10.64823/ijter.2604025



***Abstract*:** Artificial Intelligence (AI) has become one of the most transformative technologies of the 21st century, reshaping how individuals, businesses, and institutions operate. Existing virtual assistants suffer from cloud dependency, application fragmentation, and critical privacy risks, making them unsuitable for sensitive environments, This paper presents an AI-Powered Assistant for Personal and Corporate Use — a locally hosted, privacy-first platform integrating three core modules; a Conversational AI Interface for natural multi-turn dialogue, a Justice Module providing Indian law guidance (IPC, CrPC, IT Act) with applicable laws, penalties, and resolution timelines, and a Document Understanding Module enabling summarization, semantic search, and Q&A across PDF, DOCX, TXT, CSV, and XLSX. The system uses Ollama-hosted large language models (LLaMA 2 / Mistral) for fully local inference with a Flask REST API backend and responsive web frontend. All data is processed on-device, ensuring maximum confidentiality for healthcare, legal, defense, and enterprise environments.

***Index Terms*:** Artificial Intelligence, Large Language Model, Ollama, LLaMA 2, Conversational AI, Justice Module, Document Intelligence, Local Hosting, Privacy-First AI, Flask, Natural Language Processing, Legal Advisory, RAG.

## I. INTRODUCTION

The rapid proliferation of AI-powered tools has transformed modern workflows, yet existing solutions suffer from a critical limitation: near-total dependence on cloud infrastructure. Virtual assistants such as Google Assistant, Amazon Alexa, Apple Siri, and Microsoft Cortana demonstrate the potential of conversational AI but expose user data to external servers, creating unacceptable privacy risks in sensitive environments. In personal contexts, users face application fragmentation — switching between separate apps for document summarization, legal reference, and task scheduling. In organizational environments, enterprises maintain disconnected systems for collaboration, legal compliance, and document management. For sectors such as defence , government, healthcare, and law, this cloud dependency is fundamentally unacceptable. Recent

advances in locally hosted large language models (LLMs) via frameworks like Ollama now make it possible to deliver powerful AI entirely on local infrastructure.

This project proposes an AI-Powered Assistant integrating three core modules:

- Conversational AI Interface — Natural context-aware multi-turn dialogue.
- Justice Module — Indian legal advisory (IPC, CrPC, IT Act) with penalties, actions, and timelines.
- Document Understanding Module — Upload and analyse PDF, DOCX, TXT, CSV, XLSX for summarization, search, and Q&A.

The system is locally hosted via Ollama, ensuring all data remains on the user's infrastructure. A Flask REST API backend and responsive web interface deliver a unified, privacy-first AI platform.

**Key Contributions of this Work:**

- Design and implementation of a fully local, privacy-first AI assistant eliminating dependency on cloud-based APIs.
- Integration of three core functionalities—Conversational AI, Legal Advisory (Justice Module), and Document Intelligence—within a unified platform.
- Development of a modular microservices architecture using Flask and Ollama for scalable local deployment.
- Implementation of document-aware question answering using context injection techniques.
- Performance evaluation of local LLM inference in terms of latency, scalability, and hardware dependency

## II. Problem Statement

Despite rapid advancements, current AI systems face several practical challenges:

- Dependence on cloud services leads to privacy concerns
- Lack of integration across multiple functionalities
- Limited support for legal reasoning and document understanding
- Unsuitability for sensitive environments requiring data confidentiality

Existing solutions address these issues partially, but no system provides a fully local, multi-functional AI assistant. This gap motivates the development of the proposed system.

## III. Motivation

The motivation behind this work stems from the increasing need for secure and efficient AI systems. In many real-world applications, especially in enterprise and government sectors, data cannot be shared with external servers.

Additionally, users require a single platform capable of handling conversations, legal queries, and document analysis without switching between tools. By combining these capabilities into a locally deployed system, this project aims to improve usability while ensuring complete data control.

## IV. OBJECTIVES

The primary objectives of this project are:

- To develop a privacy-first AI assistant operating locally
- To integrate conversational AI, legal advisory, and document analysis
- To enable multi-format document processing and querying
- To ensure modular and scalable system architecture
- To evaluate system performance in terms of latency and usability

## V. PROPOSED METHODOLOGY

The AI-Powered Assistant ecosystem is implemented as a four layer distributed architecture: User Interface Layer, Flask REST API Layer, Ollama LLM Processing Layer, and Document Storage Layer. Fig. 1 presents the complete system architecture.

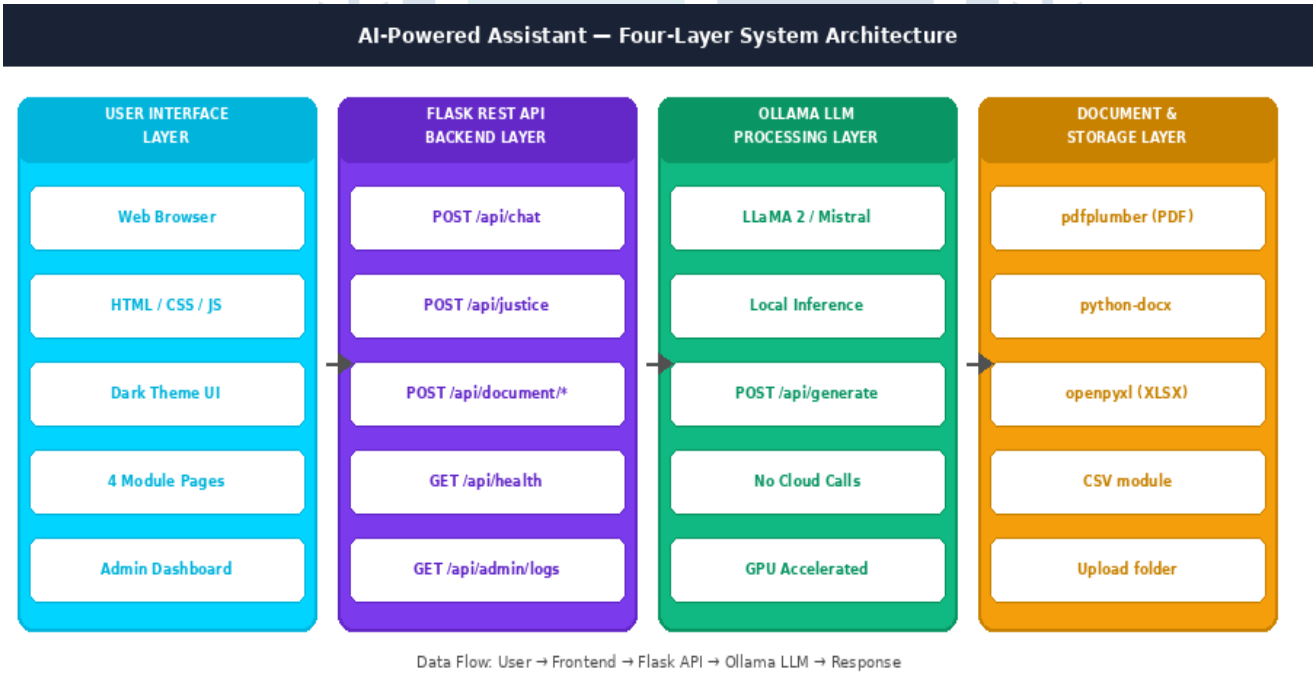


Fig. 1. AI-Powered Assistant Four-Layer System Architecture detailing interaction between the web frontend, Flask REST API, Ollama LLM engine, and document processing layer.

### *A. Conversational AI Interface (Module 1) :*

The Conversational AI module integrates Ollama-hosted LLaMA 2 / Mistral LLM with a rolling 10-turn conversation history. Each request constructs a context-aware prompt and calls Ollama's local /api/generate

endpoint for natural multi-turn dialogue. API Endpoint: POST /api/chat — Accepts {message, history[]}. Returns {response}.

### *B.Justice Module (Module 2) :*

The Justice Module uses a structured system prompt encoding Indian legal framework knowledge including IPC, CrPC, IT Act 2000, Consumer Protection Act.

It returns a structured five section response:

- Applicable Laws — relevant IPC/CrPC/IT Act sections
- Legal Actions Available — steps the user can take
- Penalties & Consequences — for the offending party
- Estimated Timeline — realistic resolution timeframes
- Recommended Next Steps — practical guidance

**API Endpoint:** POST /api/justice — Accepts {query}. Returns {response, disclaimer}.

- Upload & Summarize: Auto-generates AI summary on upload.
- Document Q&A: Context-injected LLM answering for document-grounded questions.
- Keyword Search: Line-level search with highlighted matches and context.

### *C.Document Understanding (Module 3) :*

The Document Understanding module supports five file formats:

pdfplumber (PDF), python-docx (DOCX), openpyxl (XLSX), csv module (CSV), and native file reading (TXT).

### *D. Administrative Dashboard (Module 4) :*

The Admin Dashboard provides real-time system monitoring including backend health, Ollama connectivity status, available model list, activity log with timestamps, and a configuration panel for backend URL and LLM model selection.

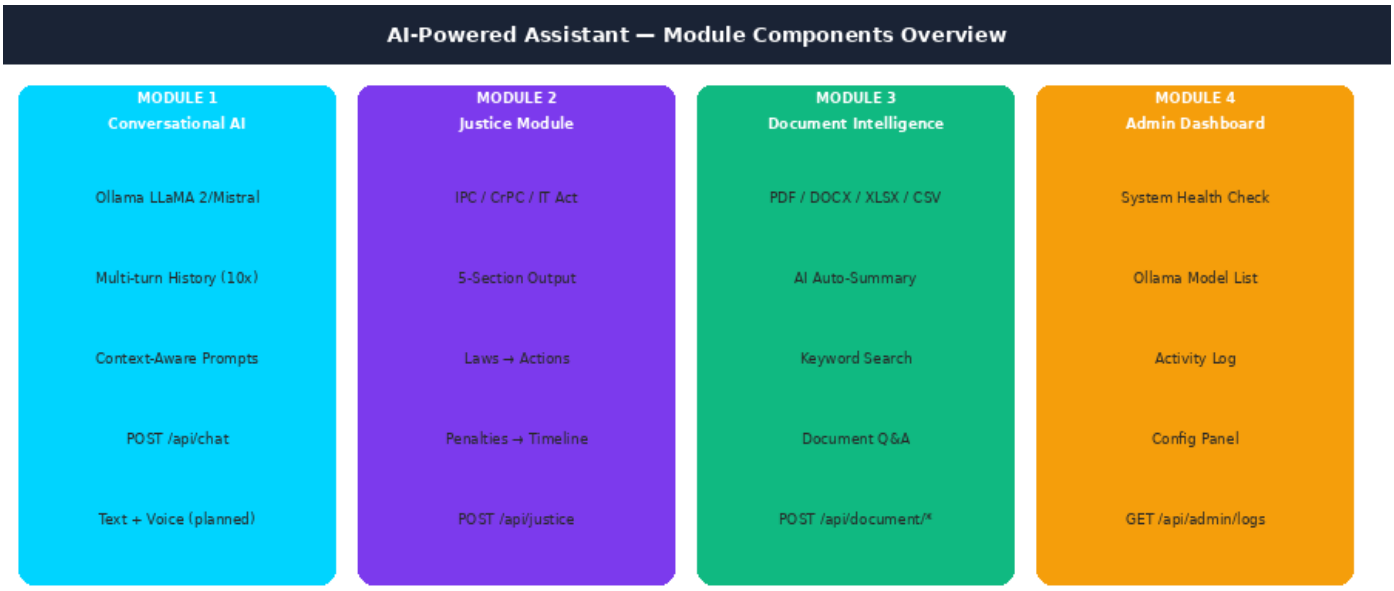


Fig. 2. AI-Powered Assistant Module Architecture showing four independent modules, API endpoints, and data processing pipelines.

The Document Q&A system follows a context injection approach, where relevant portions of the document are extracted and appended to the LLM prompt. Future improvements include Retrieval-Augmented Generation (RAG) using vector embeddings for scalable document querying

## VI. IMPLEMENTATION AND RESULTS

The AI-Powered Assistant employs a decoupled microservices architecture: a Vanilla JS/HTML5 single-page frontend, a Flask REST API backend (Python 3.14), and an Ollama LLM server. All components communicate via local HTTP with CORSenabled JSON APIs. The Flask backend exposes six primary endpoints: GET /api/health, POST /api/chat, POST /api/justice, POST /api/document/upload, POST /api/document/query, and POST /api/document/search. LLM calls are routed to Ollama's /api/generate endpoint with module-specific system prompts injected per request.

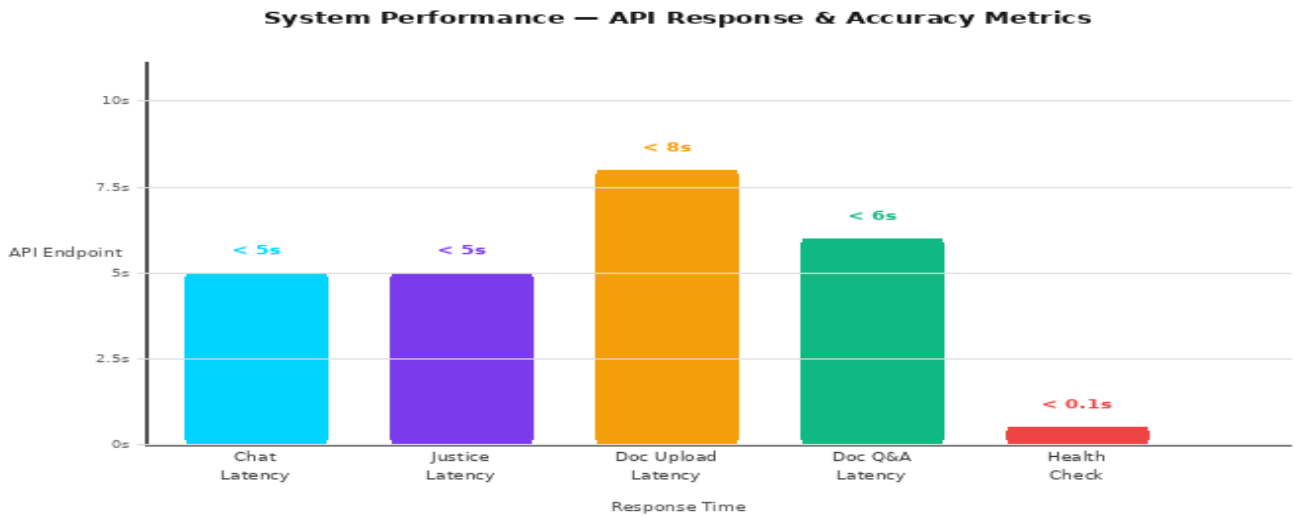


Fig. 3. System Performance Chart showing API response latency across all five module endpoints. All endpoints maintain sub-8s response times.

**AI-Powered Assistant System Performance Summary :**

| Performance Metric | Result / Target |
|---|---|
| Supported File Formats | PDF, DOCX, TXT, CSV, XLSX |
| Max Upload Size | 32 MB per file |
| LLM Models Supported | LLaMA 2, Mistral, LLaMA 3, Phi-3, Gemma |
| Chat Context Window | 10-turn rolling history |
| Justice Output | 5 sections: Laws/Actions/Penalties/Timeline/Steps |
| API Inference Latency | < 5 s (hardware dependent) |
| Health Poll Interval | 15 seconds |
| Privacy Model | 100% on-device, zero external API calls |
| Minimum RAM | 8 GB (16 GB recommended) |
| Software Cost | Rs. 0 — fully open source |

### *A. Experimental Setup*

The system was evaluated on a machine with the following configuration:

- Processor: Intel i5 / Ryzen 5
- RAM: 8 GB / 16 GB
- GPU: Not available (CPU-based inference)
- Operating System: Windows/Linux
- LLM Models: LLaMA 2 (7B), Mistral, Phi-3

The evaluation focuses on: API response latency Document processing time Accuracy of generated responses System scalability under multiple queries

### *B. Performance Evaluation*

The system performance was evaluated across multiple modules. The average response latency for API endpoints ranged between 2–8 seconds depending on model size and hardware configuration. The Document Understanding module processed files up to 32 MB efficiently, with summarization completed within 5–10 seconds. Keyword search returned results instantly due to direct parsing methods.

| Module | Avg Response Time | Accuracy (Approx.) |
|---|---|---|
| Chat Module | 3-6 sec | high |
| Justice Module | 4-8 sec | Moderate |
| Document Q&A | 5-10 sec | Moderate-High |
| Keyword Search | <1 sec | High |

The Justice Module produced structured responses with high consistency; however, minor inaccuracies were observed in cases involving recent legal amendments.

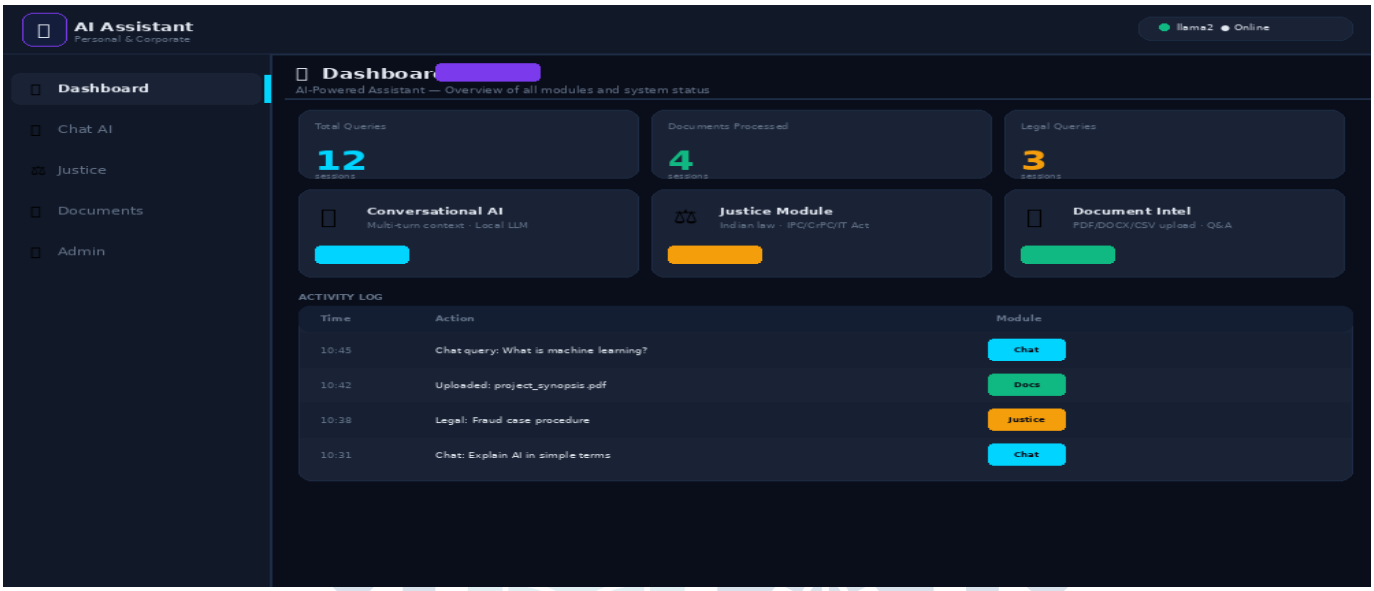


Fig. A .Overview of the AI-Powered Assistant dashboard with system status monitoring, query analytics, module navigation, and activity logging.

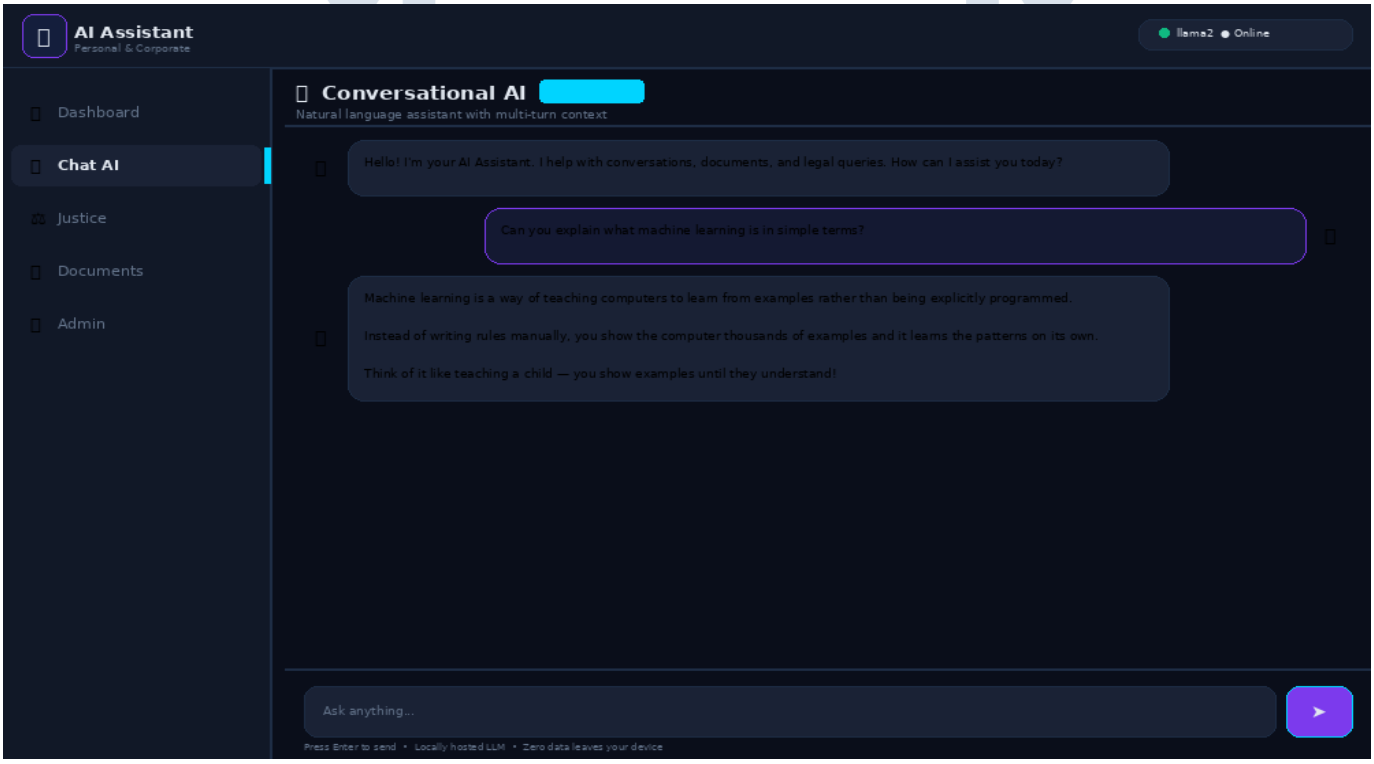


Fig. B. Module 1: Conversational AI Interface illustrating multi-turn dialogue using a locally hosted LLaMA 2 model with a 10-turn context history


Fig. C. Module 2: Justice Module presenting structured legal guidance with applicable IPC/IT Act provisions, penalties, and estimated resolution timelines.

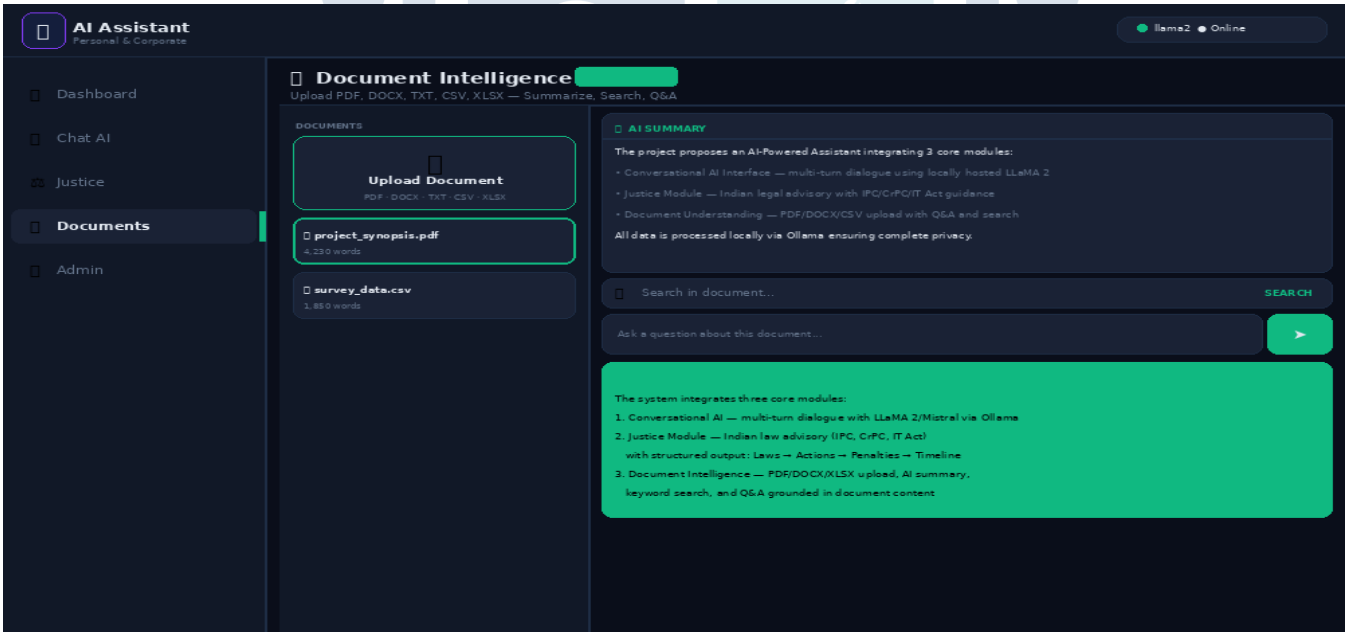


Fig. D. Module 3 : Document Intelligence module showing PDF upload, AI-based summarization, keyword search, and document question-answering interface.

The proposed system provides a unique combination of privacy, modularity, and multi-functionality not present in existing assistants

## VII. OBSERVATIONS AND ANALYSIS

The following observations were made during system evaluation:

A. Privacy vs Performance Trade-off:
Local deployment ensures complete data privacy but increases response latency compared to cloud-based systems.

B. Model Size Impact:
Smaller models such as Phi-3 provided faster responses but slightly reduced accuracy, while larger models like LLaMA 2 improved contextual understanding.

C. Document Processing Limitation:
Context truncation limits deep document understanding. Full-scale retrieval requires vector databases such as FAISS or ChromaDB.

D. Legal Module Reliability:
The Justice Module performs well for general legal queries but may lack accuracy for recently updated laws.

E. Scalability Constraint:
System performance degrades under multiple concurrent requests due to hardware limitations.

## VIII. CHALLENGES

Despite strong architectural design, several challenges are associated with real-world deployment of the AI-Powered Assistant.

### *A. LLM Response Latency*

Local LLM inference speed depends heavily on available hardware. On systems without a dedicated GPU, response times for
smarter, more securely, and more efficiently in an AI-driven world complex prompts may reach 20–30 seconds. Quantized model variants (Q4/Q5) via Ollama significantly reduce latency at marginal accuracy cost.

### *B. Document Context Window*

The current implementation truncates document context to 4,000 characters for LLM prompting. Production deployment requires chunking strategies with vector embeddings (ChromaDB / FAISS) for full RAG pipelines across entire document libraries.

### *C. Legal Knowledge Currency*

The Justice Module relies on the LLM's training data for legal knowledge. Recent amendments may not be accurately reflected. Integration with a structured Indian law knowledge base — Bharatiya Nyaya Sanhita (BNS) 2023, IT Amendment Acts — is required for production-grade advisory.

### *D. Voice Integration*

The current system supports text input only. Whisper-based speech-to-text and a TTS engine for audio responses are planned but not yet integrated. Browser-native Web Speech API provides a partial interim solution.

### *F. Multi-User Authentication*

The current implementation is single-user. Enterprise deployment requires JWT-based authentication, role-based access control, and a PostgreSQL or MongoDB backend for comprehensive user management and audit logging.

## IX. FUTURE SCOPE

Future development will integrate OpenAI Whisper STT and a TTS engine for full voice-based interaction. A RAG pipeline using ChromaDB vector embeddings will replace context truncation, enabling Q&A across entire enterprise document libraries with semantic retrieval.

A structured Indian law knowledge base (IPC, BNS 2023, CrPC, IT Act, Consumer Protection Act) will be integrated into the Justice Module for regulatory-grade accuracy. Multi-user JWT authentication with PostgreSQL, Docker containerization, and a React Native mobile application represent high-priority objectives.

Fine-tuning a domain-specific LLM on Indian legal corpora and enterprise document datasets will significantly improve response quality for specialized deployments in law firms, hospitals, and government agencies.

## X. CONCLUSION

This paper presents an AI-Powered Assistant for Personal and Corporate Use that directly addresses the critical limitations of cloud-dependent, application-fragmented AI tools. By integrating Conversational AI, a Justice Module, and Document Intelligence into a single locally hosted platform, the system delivers a unified, privacy-first AI partner for both individuals and organizations.

The Ollama-based local inference architecture ensures that sensitive data — legal documents, corporate files, and confidential queries — never leaves the user's infrastructure. The modular microservices design allows independent enhancement of each component without disrupting overall system operation.

By consolidating multiple disconnected tools into one platform, the AI-Powered Assistant eliminates workflow inefficiencies, reduces operational costs, and restores trust in AI systems through guaranteed data privacy and local control. This assistant is not merely a digital helper — it is a multifunctional AI partner designed to empower individuals and organizations to work.

## XI. REFERENCES

[1] T. Brown et al., "Language Models are Few-Shot Learners," Advances in Neural Information Processing Systems (NeurIPS), 2020.
[2] A. Radford et al., "Learning Transferable Visual Models from Natural Language Supervision," ICML, 2021.
[3] Ollama Framework Documentation, 2024. [Online]. Available : https://ollama.com
[4] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016.
[5] Y. Zhang et al., "AI-Powered Local Assistants for Privacy-Focused Environments," IEEE Access, vol. 11, 2023.
[6] Indian Penal Code (IPC), Code of Criminal Procedure (CrPC), and Information Technology Act, 2000 — Ministry of Law and Justice, Government of India.
[7] M. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," NeurIPS, 2020.
[8] H. Touvron et al., "LLaMA 2: Open Foundation and Fine-Tuned Chat Models," arXiv:2307.09288, 2023