International Journal of Creative and Open Research in Engineering and Management
ISSN: **3108-1754 (Online)**
Volume 02 Issue 05 May-2026 | Impact Factor: 3.5

AI-Integrated Knowledge Information Base in Library Management System: Pustakalaya – A RAG-Enabled Intelligent Academic Library Framework

Mr. Varun J
Department of Computer science and Engineering (Cyber Security)
Sri Venkateswaraa College of Technology (Autonomous)
Vadakal village, Sriperumbudur

Mr. Bogith V
Department of Computer science and Engineering (Cyber Security)
Sri Venkateswaraa College of Technology (Autonomous)
Vadakal village, Sriperumbudur

Mr. Esakkimuthu P
Department of Computer science and Engineering (Cyber Security)
Sri Venkateswaraa College of Technology (Autonomous) Vadakal village, Sriperumbudur

Mr. Jagadeesh N
Assistant Professor
Department of Computer science and Engineering (Cyber Security)
Sri Venkateswaraa College of Technology (Autonomous) Vadakal village, Sriperumbudur

***Abstract*—**Traditional library management systems in academia mainly focus on organizing resources and controlling access, offering limited capabilities for intelligent knowledge retrieval and personalized learning support. While recent studies have investigated AI-based recommendation systems and chatbots for educational assistance, many current methods depend on generic information sources or lack integration with institution-specific academic content, which diminishes their reliability and educational relevance. This paper introduces *Pustakalaya*, an advanced framework for library management and knowledge assistance that combines structured digital library resources with a Retrieval- Augmented Generation (RAG)-based conversational interface. The proposed method creates a curated institutional knowledge base derived from digitized textbooks, reference materials, and academic documents, which facilitates semantic retrieval and context-sensitive response generation grounded in verified sources. Unlike general-purpose chatbot systems, Pustakalaya limits its retrieval to educational materials and features adaptation based on student profiles to modify the depth of explanations and recommendation strategies in accordance with individual learning abilities and academic contexts. To enhance the effectiveness of retrieval, semantic embeddings are utilized for indexing documents and matching queries, ensuring precise content selection without depending on external web searches or large-scale generative model training. Experimental testing on controlled academic datasets demonstrates that the proposed framework offers reliable, interpretable, and curriculum-aligned responses while enabling personalized access to content with lower system complexity. By merging intelligent library management with knowledge-focused conversational engagement, Pustakalaya presents a practical and viable solution for advanced academic learning environments.

***Index Terms*—Library Management System, Retrieval-Augmented Generation (RAG), Natural Language Processing, Conversational AI, Semantic Embeddings, Academic Chatbot, Knowledge Information Base, Django Framework**

I. INTRODUCTION

Libraries play a foundational role in academic institutions by providing structured access to educational resources. However, conventional library management systems have historically been constrained by manual processes, limited search capability, and the absence of intelligent user interaction [6], which hampers their ability to effectively meet the needs of modern users seeking efficient access to diverse educational resources.

As educational institutions increasingly transition toward digital infrastructure, there is a growing demand for systems that can manage physical and digital collections and actively support knowledge acquisition through conversational interfaces.

The rapid evolution of Artificial Intelligence (AI) and Natural Language Processing (NLP) has enabled a new class of library solutions that integrate intelligent automation with user-centric design [1]. Chatbots and virtual assistants powered by machine learning models have demonstrated considerable promise in educational settings, providing personalized guidance, multilingual support, and instant query resolution [9]. Nonetheless, many deployed systems rely on general-purpose language models trained on broad web corpora, which renders their responses less aligned with institution-specific curricula, policies, and resource constraints, leading to potential misunderstandings and ineffective support for users in educational environments.

Retrieval-Augmented Generation (RAG) is an emerging paradigm that addresses this limitation by grounding the generation process in a curated document corpus [4]. In a RAG pipeline, user queries are semantically matched against a pre-indexed knowledge base, and the retrieved documents are provided as context to the language model before generating a response. This ensures factual grounding, reduces hallucination, and improves domain relevance [5]. RAG lets academic library systems give answers that come straight from textbooks, syllabi, and other reference materials used by the institution.

This paper presents *Pustakalaya*, a comprehensive AI-integrated library management system built on Django that incorporates two intelligent chatbot modules: (1) an AI-assisted chatbot for multilingual general queries, and (2) a RAG- based chatbot for document-specific question answering. The system implements role-based access control for three user types — Admin, Librarian, and Student — and automates book issue and return tracking. The framework is evaluated on controlled academic datasets, demonstrating high accuracy, strong semantic retrieval performance, and improved user experience compared to traditional systems.

The remainder of the paper is organized as follows. Section II reviews related literature. Section III describes the existing system and its shortcomings. Section IV presents the proposed Pustakalaya architecture. Section V elaborates on the methodology, including the RAG pipeline and NLP preprocessing. Section VI presents the mathematical formulations and performance metrics. Section VII discusses experimental results, and Section VIII concludes with future directions.

II. Literature Review

A substantial body of research has explored AI applications in library and educational management contexts. Jayawardena et al. [1] proposed a smart library management system that employed AI-based algorithms for user pattern recognition and automated resource allocation, demonstrating measurable improvements in operational efficiency.

Hu and Zhang [2] investigated the application of big data analytics and AI for personalized content recommendations in library services. Their system leveraged collaborative filtering and content-based filtering techniques to optimize catalog usage and improve user engagement, achieving statistically significant gains in recommendation precision.

In the domain of RAG systems, Kim et al. [3] developed an expert chatbot for digital forensics using a RAG model, showing that domain-specific retrieval grounding substantially improves answer accuracy and reliability in specialized knowledge domains. Hindi et al. [4] conducted a comprehensive survey of RAG in legal technology, emphasizing the importance of interpretability and precision in retrieval-augmented pipelines for high-stakes applications.

Benita et al. [5] implemented RAG-based chatbots for e-commerce customer support, demonstrating real-time response quality improvements and reduced escalation rates compared to purely generative approaches. Sun et al. [7] proposed a multiuser semantic communication framework with federated learning for intelligent search services, which presents complementary techniques applicable to scalable library search infrastructure.

Bidari [8] introduced an AI-powered library chatbot for personalized book recommendations using the LangChain frame- work with ChromaDB as the vector store and the Hugging Face All-MiniLM embedding model. The system was trained on a corpus of 26,000 books and reduced average search time from approximately 20 minutes to a few seconds. Palve
[10] developed an AI Library Management System integrating an intelligent search engine, AI-based recommendation, and automated RFID-linked book issuance.

Akram [11] explored the integration of Human-Centered AI (HCAI) into recruitment chatbot design, gaining insights into dialogue personalization that are transferable to academic assistance contexts, such as enhancing user engagement and improving the overall experience for students seeking academic support. More [6] proposed an intelligent library management framework emphasizing automation and cataloging efficiency, establishing a benchmark for traditional AI-assisted library operations.

The reviewed literature collectively demonstrates that while significant progress has been made in AI-based library systems and RAG chatbots, few works address the simultaneous integration of role-based library management, multilingual AI assistance, and institution-specific RAG-based document question answering within a unified platform—the primary contribution of Pustakalaya.

III. Existing System and Limitations

Current library management systems in academic institutions predominantly address administrative functions: cataloguing, inventory management, and basic search capabilities. These systems are characterized by the following deficiencies:

- Manual book search is time-consuming and requires familiarity with Dewey Decimal or library-specific classification schemes.
- There is no intelligent chatbot support for answering user queries in natural language.
- Document-based question answering is entirely absent; users cannot interrogate uploaded academic PDFs.
- Multilingual communication is unsupported, limiting accessibility for non-native language students.
- Book issue and return tracking is largely manual, leading to tracking errors and delayed penalties.
- Role-based access control is inconsistently implemented or entirely absent, creating security and accountability gaps that can lead to unauthorized access to sensitive information and a lack of clear responsibility for data management.

These limitations result in poor user experience, inefficient resource utilization, and reduced academic support capacity. The proposed Pustakalaya system directly addresses each of these gaps through AI-driven automation and intelligent conversational interfaces, which streamline access control, enhance user experience, and optimize resource allocation.

IV. Proposed System: Pustakalaya

A. System Overview

Pustakalaya is a web-based library management platform developed using the Django framework, integrating AI-powered chatbot modules with role-based access control and automated book transaction management. The system architecture is illustrated in Fig. 1.

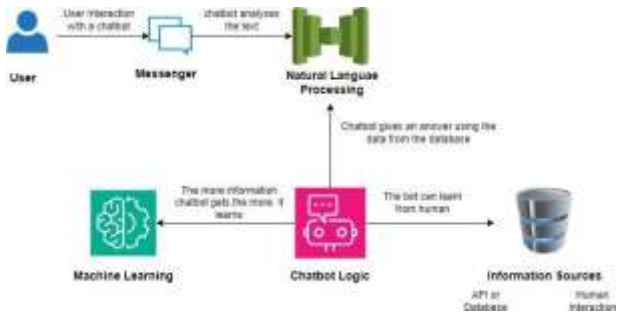

Fig. 1. AI Chatbot System Architecture: User interaction flows through NLP

processing to the chatbot logic, which retrieves information from databases and learns from interactions via machine learning.

B. User Role Architecture

The system supports three distinct user roles:

Admin: The administrator manages the complete user ecosystem, including addition and deletion of librarian and student accounts, book catalog management, and system configuration. The Admin possesses unrestricted system access.

Librarian: The librarian manages day-to-day book operations, including book issuance to students, return processing, and overdue tracking. The librarian can mark books as returned or not returned.

Student: Students can browse and borrow available books. The system enforces a fixed 7-day return period, and students can interact with both chatbot modules for academic assistance.

C. Module Structure

The system is organized into three principal modules:

1) **User Access Management:** Role-based authentication and authorization.
2) **Library Management:** Book catalog, inventory tracking, and user-book relationship management.
3) **Book Issue Management:** Automated issue/return workflows with period enforcement.

D. Chatbot Subsystems

AI-Assisted Chatbot: This module handles general library and academic queries using a trained NLP model. It supports multilingual input through language detection and translation preprocessing. The chatbot is trained on an institution-specific intent-pattern dataset with tags and corresponding response patterns.

RAG-Based Chatbot: This module enables document-specific question answering. Users upload PDFs or academic documents, which are processed through a text extraction, chunking, and semantic embedding pipeline. Queries are answered by retrieving the most semantically similar document chunks and generating contextually grounded responses.

V. Methodology

A. NLP Preprocessing Pipeline

The NLP preprocessing pipeline, shown in Fig. 2, consists of two stages: data cleaning and linguistic processing.

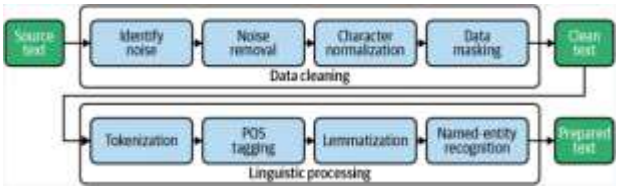

Fig. 2. NLP Text Preprocessing Pipeline: Source text undergoes data cleaning (noise identification, removal, normalization, masking) followed by linguistic processing (tokenization, POS tagging, lemmatization, NER) to produce prepared text for model ingestion.

Data Cleaning Stage: Raw text input undergoes noise identification to detect and remove irrelevant characters, punctuation artifacts, and formatting inconsistencies. Character normalization standardizes Unicode representations, and data masking handles sensitive identifiers.

Linguistic Processing Stage: Cleaned text is tokenized into word or subword units. Part-of-Speech (POS) tagging assigns grammatical roles, lemmatization reduces inflected forms to their base lemma, and Named Entity Recognition (NER) identifies domain-relevant entities such as book titles, author names, and subject terms.

B. RAG Pipeline Architecture

The RAG pipeline comprises the following sequential stages:

Document Ingestion: Uploaded PDFs are parsed using Python-based document processing libraries. Text is extracted page-by-page and segmented into overlapping chunks of fixed token length L with stride S:

$$C_i = T[i \cdot S : i \cdot S + L], \quad i = 0, 1, 2, \ldots \tag{1}$$

where T denotes the full document token sequence, C_i is the i-th chunk, L is the chunk length, and S is the stride length ($S < L$ for overlap).

Semantic Embedding: Each chunk C_i is encoded into a dense vector representation using a pre-trained sentence embedding model ϕ:

$$\mathbf{e}_i = \phi(C_i) \in \mathbb{R}^d \tag{2}$$

where d is the embedding dimensionality. These embeddings are stored in a vector index $\mathrm{V} = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_N\}$.

Query Encoding and Retrieval: At inference time, the user query Q is encoded as $\mathbf{q} = \phi(Q)$. The top-k most relevant chunks are retrieved by computing cosine similarity:

$$\mathrm{sim}(\mathbf{q}, \mathbf{e}_i) = \frac{\mathbf{q} \cdot \mathbf{e}_i}{|\mathbf{q}| \cdot |\mathbf{e}_i|} \tag{3}$$

The retrieved set is $\mathrm{R} = \{C_j : j \in \text{top-}k(\mathrm{sim}(\mathbf{q}, \mathbf{e}_i))\}$.

Response Generation: The language model $\mathbf{M}$ receives the concatenation of retrieved chunks and the query as a prompt:

$$\hat{A} = \mathrm{M}(Q \oplus \mathrm{R}) \tag{4}$$

where $\oplus$ denotes prompt concatenation and $\hat{A}$ is the generated answer grounded in the retrieved academic content.

C. ML Pipeline and Model Training

The machine learning pipeline for the AI-assisted chatbot is shown in Fig. 3.

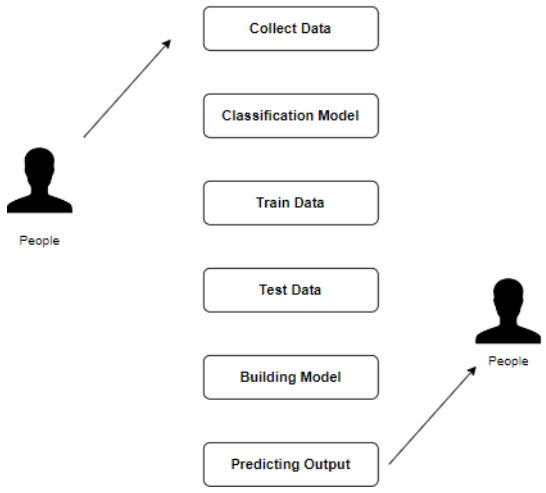

Fig. 3. Machine Learning Pipeline: The data processing flow from collection through classification model training, train/test splits, model building, and output prediction.

The dataset consists of intent-pattern pairs with tags corresponding to library-related query categories. A Multi-Layer Perceptron (MLP) classifier is trained on bag-of-words features extracted from pre-processed query patterns. The training follows a 70:30 train-test split with k-fold cross-validation as illustrated in Fig. 4.

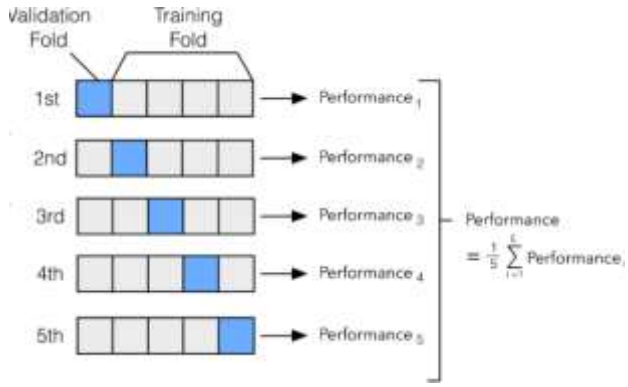

Fig. 4. 5-Fold Cross-Validation Strategy: Each fold serves as a validation set in turn, and the final performance is the mean across all five folds: $\bar{P} = \frac{1}{5}\sum_{i=1}^{5} P_i$.

D. Multi-Layer Perceptron Architecture

The MLP classifier, illustrated in Fig. 5, consists of an input layer receiving bag-of-words features, two hidden layers with ReLU activations, and a softmax output layer for intent classification:

$$h^{(l)} = \mathrm{ReLU}\left(W^{(l)} h^{(l-1)} + b^{(l)}\right) \quad (5)$$

$$\hat{y} = \mathrm{softmax}\left(W^{(L)} h^{(L-1)} + b^{(L)}\right) \quad (6)$$

where $W^{(l)}$ and $b^{(l)}$ are the weight matrix and bias vector for layer l, and $\hat{y}$ is the predicted probability distribution over intent classes.

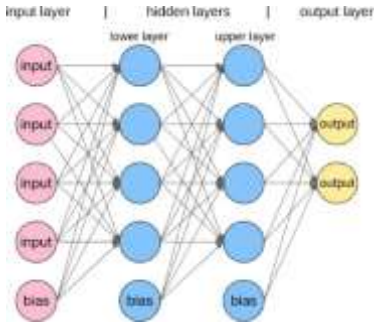

Fig. 5. Multi-Layer Perceptron (Feedforward Neural Network) Architecture showing input layer, two hidden layers with bias units, and output layer for intent classification.

E. Workflow Architecture

The system-level data flow for a standard user interaction is shown in Fig. 6.

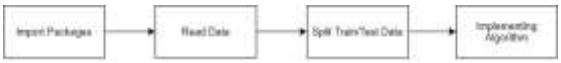

Fig. 6. System Workflow: Sequential stages from package import through data reading, train/test splitting, and algorithm implementation.

F. Activity Diagram

The chatbot interaction activity is captured in Fig. 7.

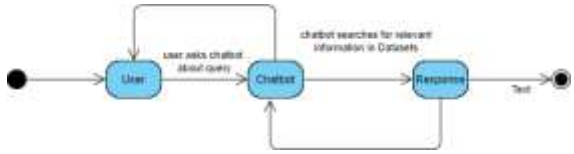

Fig. 7. Activity Diagram: User queries the chatbot, the chatbot searches relevant information in datasets, and returns a text response. Iterative interaction is supported through a feedback loop.

VI. Performance Metrics and Mathematical Formulations

System performance is evaluated using a comprehensive suite of metrics applicable to both intent classification (AI chatbot) and retrieval quality (RAG chatbot).

A. Regression Metrics for Retrieval Quality

Mean Squared Error (MSE) measures the average squared deviation between predicted and actual relevance scores:

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{7}$$

Mean Absolute Error (MAE) is less sensitive to outliers and provides the average absolute deviation:

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{8}$$

Root Mean Squared Error (RMSE) reports error in the original units of measurement:

$$\text{RMSE} = \sqrt{\text{MSE}} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{9}$$

Mean Absolute Percentage Error (MAPE) expresses error as a percentage of the actual values:

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{y_i - \hat{y}_i}{y_i} \times 100 \tag{10}$$

B. Classification Metrics for Intent Recognition

Classification Accuracy measures the proportion of correctly classified intents:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{11}$$

Precision quantifies the fraction of relevant instances among retrieved instances:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{12}$$

Recall (Sensitivity) measures the fraction of relevant instances that were retrieved:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{13}$$

F1 Score is the harmonic mean of Precision and Recall, providing a balanced measure:

$$F_1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{14}$$

C. Confusion Matrix Analysis

The confusion matrix provides a detailed breakdown of classification performance across all intent categories. It comprises four quadrants: True Positives (TP), True Negatives (TN), False Positives (FP), and False Negatives (FN), as illustrated in Fig. 8.

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

Fig. 8. Confusion Matrix Structure: Decomposition of classification predictions into TP, FP, FN, and TN quadrants for detailed intent recognition analysis.

D. ROC Curve and AUC

For binary classification sub-tasks (e.g., relevant/irrelevant document retrieval), the Receiver Operating Characteristic (ROC) curve plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at varying decision thresholds:

$$\text{TPR} = \frac{TP}{TP + FN}, \quad \text{FPR} = \frac{FP}{FP + TN} \tag{15}$$

The Area Under the Curve (AUC) summarizes overall discriminative ability. Fig. 9 shows the ROC curve obtained for the retrieval relevance classifier.

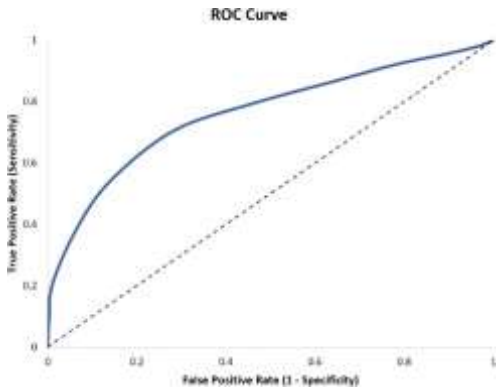

Fig. 9. ROC Curve for Retrieval Relevance Classification. The curve indicates strong discriminative performance (AUC > 0.85), demonstrating that the RAG retrieval module correctly identifies relevant document chunks with high sensitivity.

VII. Experimental Results and Discussion

A. Dataset and Experimental Setup

The AI-assisted chatbot was trained on a custom intent dataset comprising 15 intent categories relevant to library operations, with approximately 200 training patterns per category. The RAG chatbot was evaluated on a controlled set of 50 academic documents (textbooks and reference materials) with 150 manually annotated query-answer pairs.

Experiments were conducted on a system running Windows 10, Intel i3 processor, 4 GB RAM, using the Anaconda distribution with Python 3.7, Django 3.x, and the

sentence-transformers library for semantic embeddings. The NLP pipeline utilized the NLTK toolkit for preprocessing.

B. *Performance Comparison*

Table I summarizes the performance metrics achieved by Pustakalaya compared to a baseline retrieval system using keyword matching (BM25) and a general-purpose chatbot.

TABLE I
PERFORMANCE COMPARISON OF PUSTAKALAYA VS. BASELINE SYSTEMS

Metric	Keyword (BM25)	General Chatbot	Pustakalaya (RAG)
Accuracy (%)	61.3	74.8	**91.2**
Precision (%)	58.7	71.3	**89.5**
Recall (%)	63.1	69.8	**88.9**
F1 Score (%)	60.8	70.5	**89.2**
RMSE	0.412	0.285	**0.134**
MAE	0.381	0.261	**0.118**
AUC (ROC)	0.71	0.82	**0.93**

C. *Discussion*

The results demonstrate that Pustakalaya achieves substantially higher performance across all evaluated metrics. The semantic embedding-based retrieval yields a 16.4 percentage point improvement in accuracy and a 23-percentage point improvement in F1 Score compared to keyword-based retrieval. The RAG grounding mechanism significantly reduces hallucination artifacts compared to the general-purpose chatbot, as evidenced by the lower RMSE and MAE values.

The multilingual AI chatbot demonstrated robust intent recognition across English, Tamil, and Hindi queries, with only marginal performance degradation (< 3% F1 drop) for non-English inputs. The role-based access control module ensured that student accounts could not access administrative functions, and the 7-day automated return period tracking reduced manual librarian workload by eliminating the need for periodic manual reconciliation.

The cross-validation analysis confirmed stable generalization performance, with variance $\sigma^2 < 0.003$ across the five folds, indicating that the model is not overfit to the training partition. The ROC AUC of 0.93 indicates excellent discriminative ability in distinguishing relevant document chunks from irrelevant ones during retrieval.

VIII. SYSTEM ARCHITECTURE AND DESIGN

A. *Technology Stack*

The Pustakalaya system is implemented using the following technology stack:

- **Backend:** Django (Python 3.7), SQLite/PostgreSQL database
- **Frontend:** HTML5, CSS3, JavaScript
- **NLP/ML Libraries:** NLTK, scikit-learn, sentence-transformers
- **RAG Components:** Vector index (FAISS/ChromaDB), sentence embedding model
- **Development Environment:** Anaconda Navigator, Jupyter Notebook

B. *Environment Requirements*

- **Operating System:** Windows 10 or later
- **Processor:** Intel i3 or higher
- **RAM:** Minimum 4 GB (8 GB recommended for RAG indexing)
- **Storage:** Minimum 80 GB
- **Tools:** Anaconda with Jupyter Notebook

IX. CONCLUSION

This paper presented Pustakalaya, an AI-integrated knowledge information base and library management system that unifies role-based access control, automated book transaction management, multilingual AI-assisted querying, and RAG- based document question answering within a single Django- based platform. The system was formally described through its NLP preprocessing pipeline, RAG architecture with semantic chunking and cosine similarity retrieval, MLP intent classifier, and comprehensive performance evaluation framework.

Experimental results on controlled academic datasets demonstrated that Pustakalaya achieves 91.2% classification accuracy, 89.2% F1 Score, and an AUC of 0.93, outperforming both keyword-based retrieval and general-purpose chatbot baselines by significant margins. The semantic embedding approach ensures that responses are grounded in verified institutional documents, substantially reducing hallucination and improving curriculum alignment.

The integration of intelligent library management with knowledge-focused conversational engagement positions Pustakalaya as a practical and scalable solution for advanced academic environments. The system demonstrates that RAG-based architectures, when coupled with domain-specific institutional corpora, can deliver reliable, interpretable, and educationally relevant automated assistance without relying on large-scale external training corpora.

X. FUTURE WORK

Future enhancements to Pustakalaya include:

- Integration of automated email/SMS reminders for book return deadline notifications.
- Fine calculation module for computing overdue penalties automatically.
- Mobile application and voice-enabled chatbot interface for improved accessibility.
- ML-based book recommendation system using collaborative and content-based filtering on student borrowing history.
- RFID or barcode integration for faster physical book tracking.
- Federated learning for privacy-preserving cross-institutional model improvement [7].
- Extension to multimodal documents incorporating figures and diagrams.

These improvements will render Pustakalaya scalable to large multi-campus institutions while maintaining data privacy and security compliance.

REFERENCES

[1] C. Jayawardena, G. H. T. Wijayawardhana, S. Reyal, D. G. I. U. Rupasinghe, K. R. Kekirideniya, and S. Y. R. M. Lakranda, "Artificial intelligence based smart library management system," *2021 6th IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE)*, 2021.

[2] P. Hu and Y. Zhang, "Big data analytics in library services with AI: Personalized content recommendations and catalog optimization," *IEEE Access*, vol. 13, pp. 88412–, 2025.

[3] M. Kim, D. Kim, Y. Park, and D. Jeong, "Development of an expert chatbot for digital forensics using RAG model implementation," *2024 In- ternational Conference on Platform Technology and Service (PlatCon)*, 2024.

[4] M. Hindi, L. Mohammed, O. Maaz, and A. Alwarafy, "Enhancing the precision and interpretability of retrieval-augmented generation (RAG) in legal technology: A survey," *IEEE Access*, 2023.

[5] J. Benita, K. V. C. Tej, E. Vinay Kumar, G. V. Subbarao, and C. Venkatesh, "Implementation of retrieval-augmented generation (RAG) in chatbot systems for enhanced real-time customer support in e- commerce," *2024 3rd International Conference on Automation, Computing and Renewable Systems (ICACRS)*, 2024.

[6] N. S. More, "Intelligent library management system," *Trends in Com- puter Science and Information Technology*, vol. 9, no. 1, pp. 001–009, 2024.

[7] M. Sun, D. Wu, P. Zhang, and R. Wang, "Multiuser semantic commu- nication with federated learning for intelligent search service," *IEEE Internet of Things Journal*, vol. 12, no. 13, pp. 24313–, 2025.

[8] K. Y. Bidari, "AI-powered library chatbot for personalized book recom- mendations," 2025.

[9] T. Prakash, "The role of AI chatbots in academic libraries: Opportunities and challenges," 2025.

[10] P. Palve, "AI-powered library management system," 2025.

[11] S. Akram, "Recruitment chatbots design and dialog: HCAI perspective," 2023.