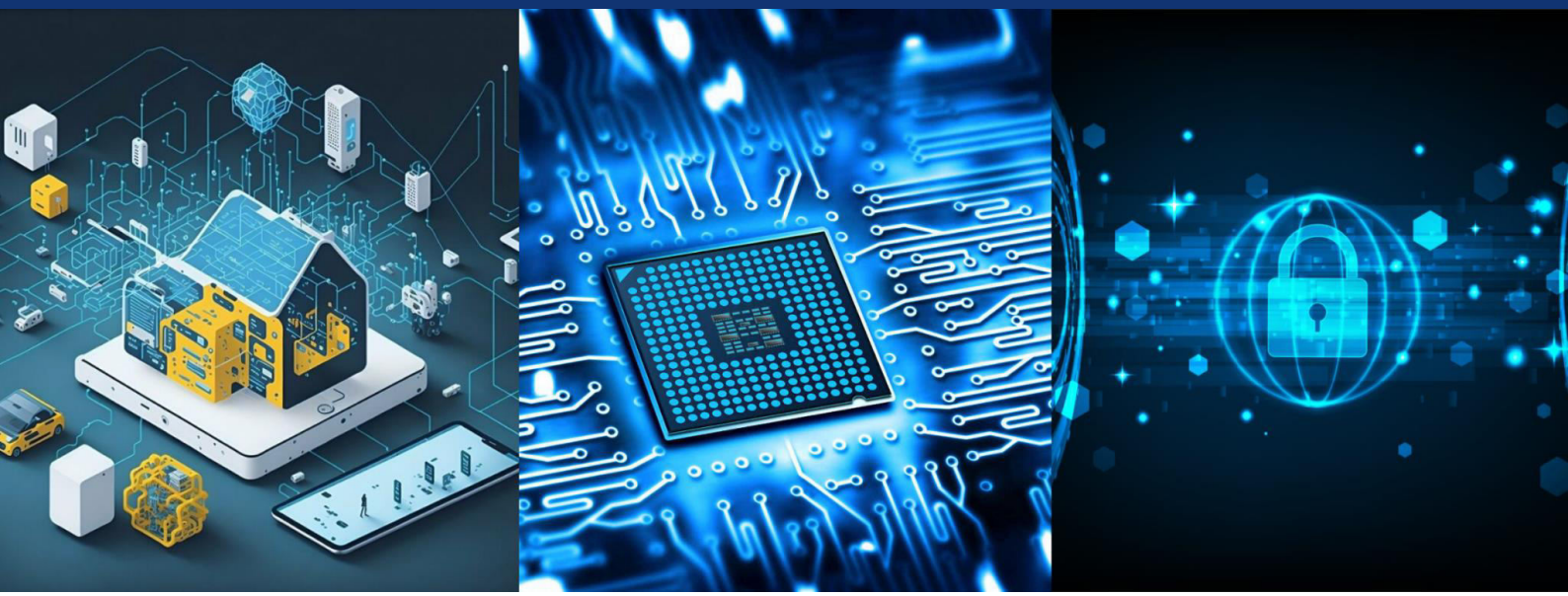
IJIRCCE

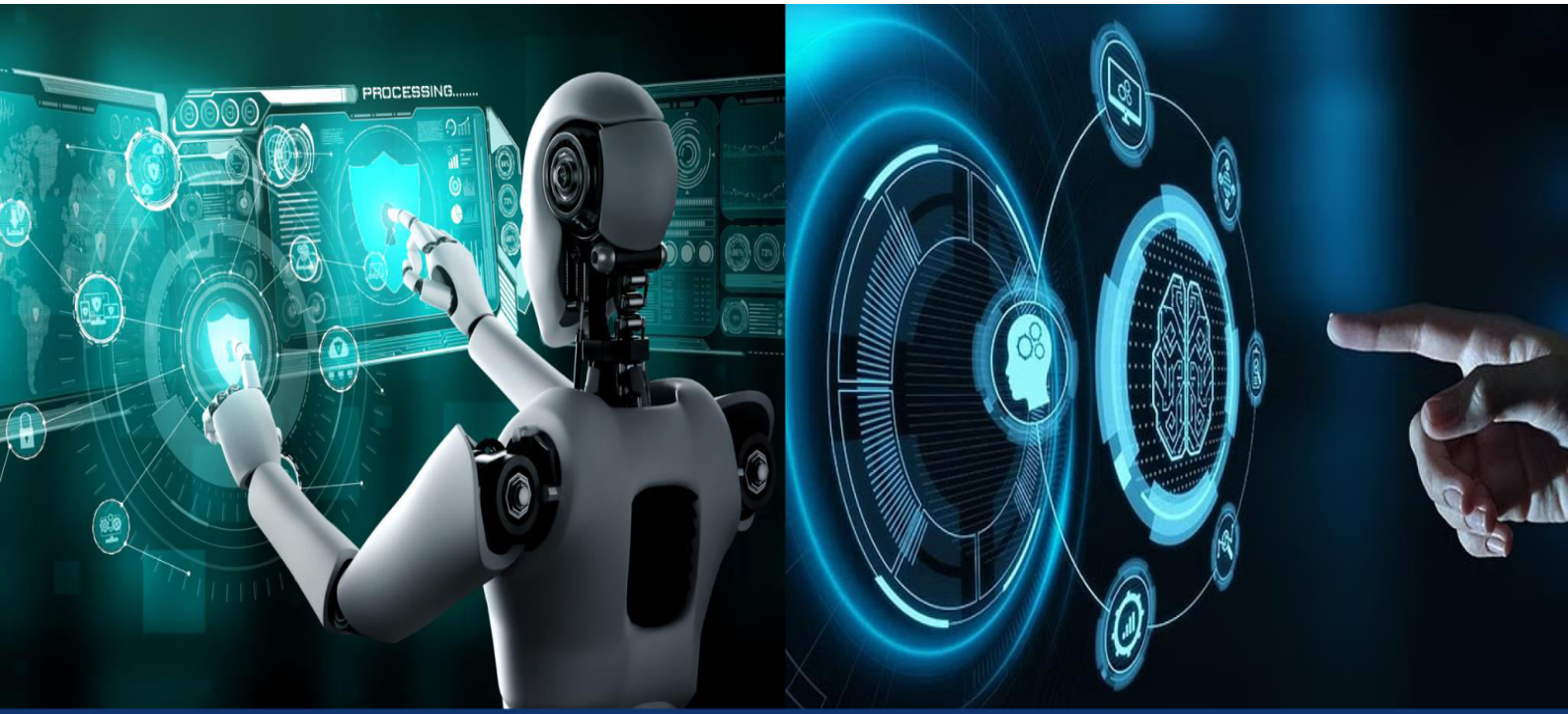
ISSN(O): 2320-9801

ISSN(P): 2320-9798



International Journal of Innovative Research in Computer and Communication Engineering

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Impact Factor: 9.927

Volume 14, Issue 10, October 2026

© 2026 IJIRCCE | Volume 14, Issue 10, October 2026 DOI: 10.15680/IJIRCCE.2026.1410026

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

CollegeBot: A Virtual Assistant System for College Enquiry using Natural Language Processing

Prajwal C A, Abhishek Hasabi, Channaveergouda Patil, Pranay K Byadgi, Er. Bhavana S Patil

UG Students (VII Semester), Department of Computer Science and Engineering, STJ Institute of Technology, Ranebennur, Karnataka, India

Assistant Professor, Department of Computer Science and Engineering, STJ Institute of Technology, Ranebennur, Karnataka, India

ABSTRACT: Colleges receive a large number of repetitive enquiries about courses, fees, eligibility, duration and admission, which are traditionally answered manually by office staff during working hours. This paper presents CollegeBot, a web-based virtual assistant that answers such enquiries in natural language. The system is built with Python and Flask for the backend, HTML, CSS and JavaScript for the chat interface, and MySQL for storing users, intents, FAQs and courses. User questions are cleaned using tokenization, stop-word removal and Porter stemming, represented with TF-IDF features (unigrams and bigrams), and matched with stored FAQ questions using cosine similarity. A Multinomial Naive Bayes model is trained over the same features to model the intent categories. The chatbot recognises course names, answers questions that mention more than one course, and returns a fallback message when the best similarity falls below a threshold of 0.35, so that unrelated answers are avoided. Role-based access control separates students from administrators, who maintain intents, courses and FAQs through protected pages. Functional testing of the main modules confirmed correct behaviour for login, greetings, course queries, administrative updates and unsupported questions.

KEYWORDS: Chatbot, Natural Language Processing, TF-IDF, Cosine Similarity, Naive Bayes, Flask, MySQL, College Enquiry, Virtual Assistant.

I. INTRODUCTION

Students and parents regularly need quick information about a college: which courses are offered, how long they last, what they cost, who is eligible and how to apply. In the traditional process this information is obtained by visiting the college office, telephoning staff or searching several web pages. The process is slow, depends on staff availability and may produce inconsistent answers when different people reply to the same question.

Chatbots based on Natural Language Processing (NLP) offer a practical alternative because the user simply types a question such as “What is the fee for BCA?” or “How can I apply for admission?” and receives an immediate reply. CollegeBot is a web-based virtual assistant developed for this purpose. It stores college information in a MySQL database, understands questions using NLP techniques, and returns a suitable stored answer or a safe fallback message. The main contributions of this work are: (i) a complete, low-cost college enquiry chatbot that runs on a standard Flask and MySQL stack; (ii) a hybrid matching pipeline that combines TF-IDF, cosine similarity and course-name detection; (iii) support for questions that mention several courses; and (iv) an administrator module with role-based access so that the knowledge base can be updated without changing code.

II. LITERATURE SURVEY

Several researchers have applied NLP to student support and admission assistance. Singhuang and Natho [1] proposed an enhanced chatbot for university admission and educational guidance. Their system uses a structured knowledge base covering admission procedures, eligibility, courses, fee structures, scholarships and deadlines, and applies machine learning so that response accuracy improves with use.

© 2026 IJIRCCE | Volume 14, Issue 10, October 2026 DOI: 10.15680/IJIRCCE.2026.1410026

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Mallari et al. [2] developed DHViChat for an admissions office. The system applies tokenization, intent recognition and entity extraction, and consists of a user interface, an NLP engine and a backend database. Queries pass through a translation service and a user-defined filter before a language model generates the reply.

Abdul et al. [3] presented a college assistance chatbot that combines NLP with an Artificial Neural Network (ANN). The knowledge base covers courses, admissions, fees and examination schedules, and the ANN is trained on student queries to classify them. Queries that cannot be answered with sufficient confidence are stored as unanswered questions so that the curated database can be extended.

These studies show that a curated knowledge base with intent recognition is effective for enquiry handling. However, neural and large-language-model approaches need training data, computing resources or external services, which may be unsuitable for a small institution. CollegeBot therefore uses lightweight classical NLP methods [4]–[6] with an editable database, which keeps the system simple, transparent and easy to maintain.

III. EXISTING AND PROPOSED SYSTEM

A. Existing System

In the existing process, administrative staff provide information through office visits, telephone calls and other manual channels. This increases staff workload, causes delays, is unavailable outside office hours, produces long waiting times during admission periods and may give inconsistent answers because responses depend on individual staff members.

B. Proposed System

The proposed system is an AI-based chatbot accessed through a web browser. It offers quick responses to frequently asked questions, reduces the load on staff, understands natural-language input, handles questions about admission, fees, eligibility, courses and duration, supports queries containing more than one course, and gives a fallback response when the required information is not stored.

Table I compares the surveyed systems with CollegeBot.

TABLE I. COMPARISON OF RELATED WORK

System	Technique	Remark
[1] Admission chatbot	NLP, machine learning	Knowledge base for admission, fees, deadlines
[2] DHViChat	Tokenization, intent, entities, language model	Needs external translation and language-model services
[3] ANN chatbot	NLP + ANN classifier	Needs a training dataset; logs unanswered queries
CollegeBot	TF-IDF, cosine similarity, Naive Bayes, MySQL	Lightweight, editable by admin, safe fallback

IV. OBJECTIVES AND REQUIREMENTS

4A. Objectives

- **Round-the-clock support:** answer user queries at any time without human assistance, reducing staff workload.
- **Role-based access control:** allow students and administrators to access only the functions authorised for their role.
- **Quick information:** give students and faculty fast and consistent answers on admission, fees, eligibility, courses and duration.
- **Query understanding:** analyse the quality and meaning of the user message using NLP and respond promptly.

B. Functional Requirements

The system shall accept questions from the user, pre-process the text and identify important words, identify the intent (admission, fees, eligibility, course details or duration), retrieve the answer from MySQL, return a fallback response

© 2026 IJIRCCE | Volume 14, Issue 10, October 2026 DOI: 10.15680/IJIRCCE.2026.1410026

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

when no suitable information exists, allow users to register and log in, and allow administrators to manage intents, courses and FAQs.

C. Non-Functional Requirements

The main non-functional requirements are performance (quick replies to normal queries), scalability (correct operation with multiple users), usability (a simple interface), reliability (appropriate answers), availability (whenever the Flask application and MySQL server are running) and maintainability (updates through the management pages).

D. Hardware and Software Requirements

TABLE.II. HARDWARE AND SOFTWARE REQUIREMENTS

Component	Specification
Processor / RAM	Intel i5 or equivalent / minimum 4 GB
Storage / System	256 GB SSD / 64-bit computer
Operating system	Windows 11
Front end	HTML, CSS, JavaScript
Back end	Python, Flask
NLP / ML libraries	NLTK, scikit-learn
Database	MySQL (run through XAMPP)
Tools	VS Code, Google Chrome

V. SYSTEM DESIGN

A. Architecture

The architecture has six components, shown in Fig. 1. The user enters a question in the chat interface. The input module sends it to the Flask API, which passes it to the NLP module. The NLP module pre-processes the text, finds the best matching FAQ and consults the MySQL database, which holds the Users, Intents, FAQs and Courses tables. The output module formats the answer, and the web interface displays it.

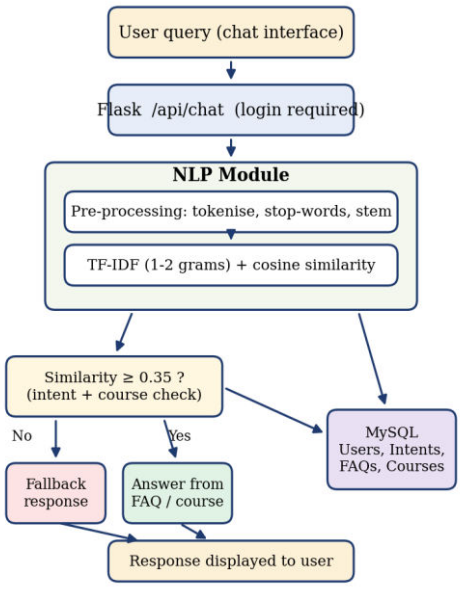


Fig.1. Processing flow of CollegeBot

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

B. Development Methodology

The Waterfall model was followed in six phases: requirement analysis, system design, implementation, testing, deployment and maintenance. Each phase was completed before the next began, which suited the clearly defined scope of the project. XAMPP is used to run the MySQL server in the local environment.

C. Data Flow and Interaction

The data flow has three processes. Process 1.0 (Process Query) receives the question and performs NLP pre-processing. Process 2.0 (Intent Classification) identifies the intent using TF-IDF features. Process 3.0 (Fetch Response) reads the relevant FAQ or course information from the database and prepares the reply, or the fallback message if nothing suitable is found.

The interaction between modules follows eight steps: the user sends the query to the web interface; the interface forwards it to the input module; the input module passes it to the NLP module; the NLP module checks the intent in the database; the database returns the result; the NLP module generates the response; the output module sends it to the web interface; and the interface displays it to the user.

D. Use Cases

Two actors use the system. The User logs in, asks questions on admission, fees, eligibility, courses, duration or general FAQs, and receives a reply or a fallback message. The Admin logs in through a separate admin login and manages intents, FAQs, courses and admission information.

E. Database Design

The database has four tables (Table III). Each FAQ belongs to exactly one intent and optionally to one course. A NULL course identifier marks a general FAQ, such as a greeting or an admission procedure that applies to all programmes. Passwords are stored as salted hashes rather than plain text.

TABLE.III. DATABASE TABLES

Table	Main columns
users	user_id (PK), name (unique), password (hash), role (user/admin)
intents	intent_id (PK), intent_name (unique)
courses	course_id (PK), course_name (unique), duration, fees, eligibility
faqs	faq_id (PK), intent_id (FK), course_id (FK, NULL allowed), question, answer

VI. IMPLEMENTATION

A. Technology Stack

The backend is written in Python with the Flask framework. NLTK supplies the Porter stemmer [5] and scikit-learn supplies the TF-IDF vectoriser, cosine similarity and Multinomial Naive Bayes classifier [6]. The front end uses HTML, CSS and JavaScript, and a small script sends each question to the /api/chat route using the fetch API and shows the returned answer. MySQL is accessed through the mysql-connector library.

B. Text Pre-processing

Every question is converted to lower case and split into alphanumeric tokens. Common stop words such as “what”, “is”, “the”, “for” and “about” are removed, and each remaining token is reduced to its stem. For example, “What is the fee for BCA?” becomes the compact form “fee bca”. The same function is applied to the stored FAQ questions, so that the user input and the knowledge base are compared in the same form.

C. Intent Recognition and Answer Matching

The pre-processed FAQ questions are converted to TF-IDF vectors using unigrams and bigrams [7]. The weight of a term t in a document d is $w(t,d) = tf(t,d) \times \log(N / df(t))$, where N is the number of FAQ questions and df(t) is the number of questions containing t. A Multinomial Naive Bayes classifier is trained on these vectors with the intent names as class labels. For answer matching, the user question is vectorised in the same way and compared with every stored question using cosine similarity:

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

$$\text{sim}(q, f) = (q \cdot f) / (\|q\| \|f\|) \quad (1)$$

The stored question with the highest similarity is selected and its intent is taken as the matched intent. If the highest similarity is below the threshold of 0.35, the system rejects the match and returns the fallback message instead of an unrelated answer.

D. Course Detection and Multi-Course Queries

Course names are read from the database and compared with the question after both are normalised (lower case, punctuation removed), so that written forms such as "B.Com" are handled consistently. If the matched intent is course-specific (all its FAQs belong to a course) and the user has not mentioned any course, the system returns the fallback message rather than guessing. When one course is mentioned, the answer is taken from the FAQ of that intent and course. When two or more courses are mentioned, the best-matching FAQ is found separately for each course and the answers are joined, so that a question such as "fee of BCA and BBA" is answered in a single reply.

E. Algorithm

- **Step 1:** Receive the question from the chat interface and reject empty input.
- **Step 2:** Pre-process the question (tokenize, remove stop words, stem).
- **Step 3:** Detect course names mentioned in the question.
- **Step 4:** If two or more courses are found, retrieve the best FAQ for each course and combine the answers.
- **Step 5:** Otherwise compute TF-IDF cosine similarity with all stored questions and select the best one.
- **Step 6:** If similarity < 0.35, or a course-specific intent has no course, return the fallback message.
- **Step 7:** Retrieve the answer from MySQL and send it to the interface as JSON.

F. Authentication and Administration

Registration and login use hashed passwords, and the user role is stored in the session after login. Two decorators protect the routes: one requires a logged-in user and the other requires the admin role. This provides role-based access control (RBAC): students can only use the chat page, whereas administrators can open the dashboard and add or delete intents, courses and FAQs. After every change the chatbot reloads its data, so that new knowledge is available immediately without restarting the application.

G. Worked Example

Consider the question "What is the fee for BCA and BBA?". (1) Pre-processing removes the stop words and stems the remaining tokens, giving "fee bca bba". (2) Course detection compares the normalised question with the course names stored in the database and finds two courses, BCA and BBA. (3) Because two courses were found, the system looks at the FAQs of BCA and of BBA separately. (4) For each course, the cosine similarity between the question vector and the FAQ question vectors is computed, and the best FAQ is chosen only if its score is at least 0.35. (5) The two answers are joined with the course name as a prefix, and the combined text is sent to the chat page as JSON together with the matched intent and a confidence value.

H. Administrator Module

The dashboard shows the current number of users, intents, FAQs and courses. The intent page lists all intents with add and delete options. The course page contains a form for the course name, duration, fees and eligibility. The FAQ page has a form with an intent list, a course list that includes a General / No Course option, a question box and an answer box, followed by a table of all stored FAQs. Foreign-key constraints protect data integrity: an intent that is used by FAQs, or a course that is used by FAQs, cannot be deleted, and the administrator is shown an explanatory message.

VII. RESULTS AND DISCUSSION

The application provides a home page with user login, admin login and registration, a chat page titled "College Enquiry Assistant", and an administrator dashboard showing the number of users, intents, FAQs and courses together with pages for managing intents, courses and FAQs. The demonstration database contains the courses BCA, BBA and B.Com, with FAQs on admission, fees, eligibility, duration, course list, greetings and thanks.

© 2026 IJIRCCE | Volume 14, Issue 10, October 2026 DOI: 10.15680/IJIRCCE.2026.1410026

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

The system was tested using the functional test cases in Table IV. All the cases produced the expected result.

TABLE.IV. TEST CASES AND RESULTS

ID	Input / action	Result
TC01	Open the website	Home page shown
TC02	Login with valid credentials	Chatbot accessible
TC03	Course-related question	Course information shown
TC04	Ask about course fees	Fee shown
TC05	Ask about duration	Correct duration shown
TC06	Greeting “Hello”	Greeting reply shown
TC07	Unsupported question	Fallback shown
TC08	Add/retrieve FAQ as admin	Stored/retrieved correctly
TC09	Access MySQL database	Tables and data available

Table V shows sample questions and the behaviour expected from the demonstration database.

TABLE V. SAMPLE QUERIES AND RESPONSES

User query	System behaviour
What is the fee for BCA?	Returns the stored BCA fee answer (course-specific fees intent)
How long is B.Com?	Returns the 3-year duration answer for B.Com
Eligibility for BBA?	Returns the BBA eligibility answer
How can I apply for admission?	Returns the general admission answer
What courses are available?	Lists BCA, BBA and B.Com
Fee of BCA and BBA	Joins the fee answer of each course in one reply
Hello	Greeting reply
Who won the cricket match?	Fallback message

The fallback behaviour is an important design choice. Because a similarity threshold and a course check are applied before an answer is returned, the chatbot answers only from its stored knowledge and avoids misleading replies. The administrator module makes maintenance simple, since a new course or FAQ can be added through a form. The tests were functional; a large-scale accuracy evaluation with real user queries has not yet been carried out and is planned as future work.

VIII. LIMITATIONS, CONCLUSION AND FUTURE SCOPE

The current system has some limitations. It answers only questions that are close to the stored FAQ questions, so its coverage depends on how many FAQs the administrator has entered. It does not keep the context of earlier messages in a conversation, supports only English text, and the demonstration data uses sample fees and eligibility values. In addition, the Multinomial Naive Bayes model is trained on the FAQ data to represent the intent categories, while the final answer selection depends mainly on the similarity score and the course check.

CollegeBot provides an easy and interactive way to obtain college information. By combining a Flask web application, classical NLP techniques and a MySQL knowledge base, it answers course, fee, eligibility, duration and admission questions, handles multiple courses in one query, and falls back safely when no suitable answer exists. Role-based access allows authorised administrators to keep the information up to date.

The system can be extended by adding more intents and programmes, support for regional languages such as Kannada, voice input, an unanswered-question log to guide administrators, edit options for existing records, and larger datasets

www.ijircce.com | e-ISSN: 2320-9801, p-ISSN: 2320-9798| Impact Factor: 9.927| ESTD Year: 2013|

International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

for measuring precision and recall. Integration with messaging platforms and the college website would make the assistant available to more users.

IX. ACKNOWLEDGEMENT

The authors thank the Department of Computer Science and Engineering, STJ Institute of Technology, Ranebennur, for providing the facilities and guidance needed to carry out this work.

REFERENCES

1. N. Singhuang and P. Natho, “An Enhanced Chatbot for University Admission and Educational Guidance,” International Journal of Engineering and Advanced Technology (IJEAT), 2025.
2. C. S. Mallari, P. I. Docto and M. A. Cruz, “DHViChat: An Intelligent Chat Assistant for the Admissions Office using Natural Language Processing,” International Journal of Advanced Computer Science and Applications (IJACSA), 2024.
3. M. S. Abdul, M. V. Manoj Kumar, C. R. Nirmala and R. Naseer, “Enhancing Student Support with AI: A College Assistance Chatbot Using NLP and ANN,” International Journal of Innovative Research in Computer Science and Engineering, 2024.
4. S. Bird, E. Klein and E. Loper, Natural Language Processing with Python, O’Reilly Media, 2009.
5. M. F. Porter, “An algorithm for suffix stripping,” Program, vol. 14, no. 3, pp. 130–137, 1980.
6. F. Pedregosa et al., “Scikit-learn: Machine Learning in Python,” Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.
7. G. Salton and C. Buckley, “Term-weighting approaches in automatic text retrieval,” Information Processing & Management, vol. 24, no. 5, pp. 513–523, 1988.
8. Pallets Projects, “Flask Documentation.” [Online]. Available: https://flask.palletsprojects.com/
9. Oracle, “MySQL Reference Manual.” [Online]. Available: https://dev.mysql.com/doc/



INTERNATIONAL STANDARD SERIAL NUMBER INDIA

INNO SPACE SJIF Scientific Journal Impact Factor

doi crossref

निस्केयर NISCAIR

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

9940 572 462 6381 907 438 ijircce@gmail.com

www.ijircce.com

Scan to save the contact details