VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama ,Belagavi–590018

Report On

“Smart Enquiry Chatbot”

Bachelor of Engineering in Artificial Intelligence & Machine Learning

Submitted by

PRATHEEK B K

(1AM22AI040)

V PRUTHVIRAJ GOWDA

(1AM22AI056)

Under the Support and Guidance of

Prof. K B Bini

AMC ENGINEERING COLLEGE

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

18th K.M.Bannerghatta Main Road, Bengaluru-560083

2025-26

Introduction:

In many educational institutions, enquiry and information services are traditionally handled by administrative staff through telephone calls and emails. This often leads to delays in response, repetitive work, and increased workload for the staff. Students, parents, and visitors frequently ask similar questions regarding admissions, fee structure, placements, courses, hostel facilities, and other campus-related information.

To overcome these limitations and improve efficiency, the use of Artificial Intelligence (AI) based automation has become essential. A **Smart Enquiry Chatbot** can provide instant responses to commonly asked queries, operate 24/7, reduce manual effort, and improve overall user experience.

The objective of this project is to design and develop a **web-based AI-powered University Enquiry Chatbot** capable of answering frequently asked questions (FAQs) in real-time and escalating complex or unknown queries to the administrative team for further support.

Project Description:

Overview

The Smart Enquiry Chatbot is an interactive web application that allows users to communicate through a chat interface and receive instant responses. It leverages Natural Language Processing (NLP) techniques to understand user queries and retrieve the most relevant answers from a predefined FAQ knowledge base.

The system is lightweight, fast, and deployable within a university environment without requiring heavy computational resources or internet-dependent APIs.

Features of the System

Feature	Description
Real-time chat	Users can ask university-related queries and receive instant responses.
FAQ knowledge base	Contains curated answers regarding admissions, fees, courses, placements, events, etc.
NLP-based matching	Uses TF-IDF and cosine similarity to find the best answer.
Auto-escalation	If confidence is low, the query is forwarded to admin for review.
Typing delay simulation	Adds human-like delay before sending response for better UI feel.
Web interface	Accessible through browser on desktop or mobile.
Easy to update	Admin can add new FAQs to improve chatbot learning.

System Architecture

The architecture follows a Client-Server model:

1. Frontend (Client):
    - HTML, CSS & JavaScript
    - Chat UI + typing animation
    - AJAX request for every user query
    - Auto-scroll for smooth interaction
2. Backend (Server):
    - Flask (Python web framework)
    - NLP processing using TF-IDF Vectorizer
    - Similarity calculation using Cosine Similarity
    - Handles query escalation and JSON storage
3. Database / Knowledge Base:
    - FAQ dataset stored as JSON file
    - Escalated queries stored separately for future FAQ updates

Workflow of the System

1. User enters a query in the chat interface
2. Query is sent to backend API (/api/chat)
3. Text preprocessing and similarity matching performed
4. If similarity score ≥ defined threshold → Return best FAQ answer
5. Else → Response with message that query is forwarded to admin and stored in escalation file
6. Chat interface displays bot reply with typing animation
7. Auto-scroll adjusts to show the latest conversation

Tools & Technologies Used

Component	Technology
Programming Language	Python
Framework	Flask
NLP Library	Scikit-learn (TF-IDF, Cosine Similarity)
Frontend	HTML, CSS, JavaScript
Storage	JSON Files
Development Environment	VS Code

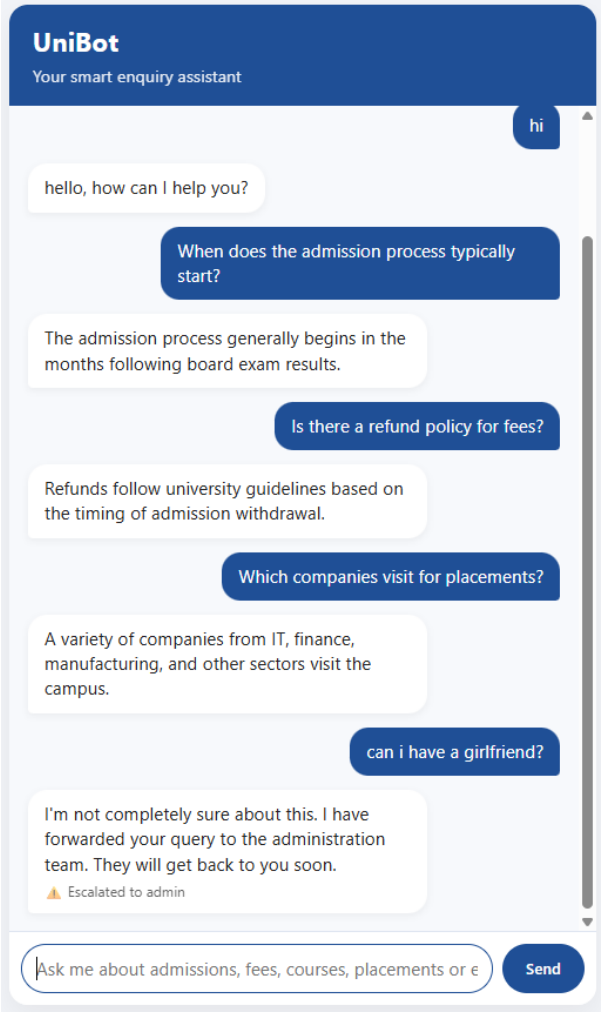
Source code:

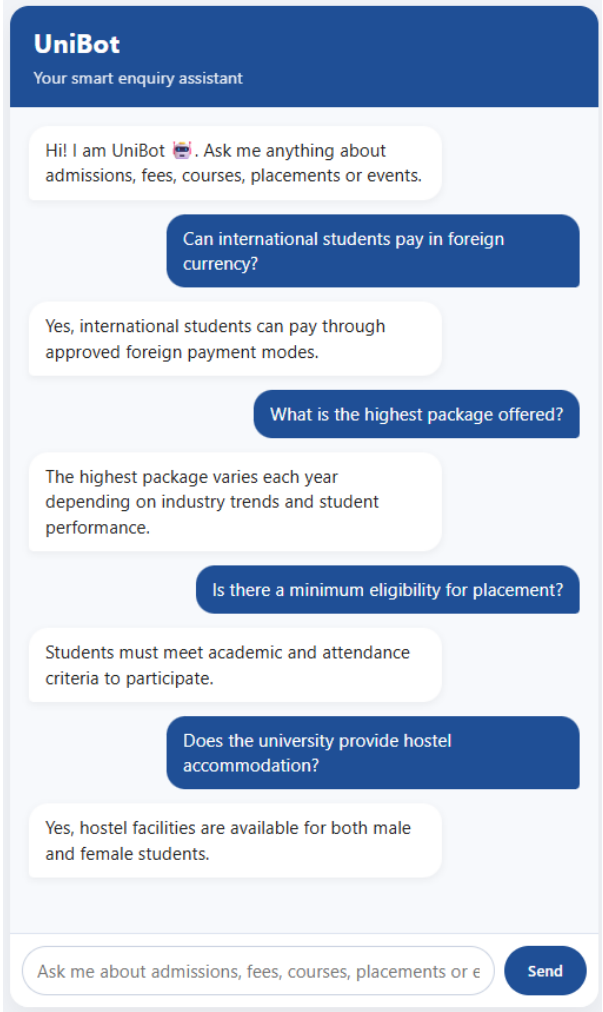
```python
FAQ_FILE = "faq.json"
ESCALATION_FILE = "escalated_queries.json"

with open(FAQ_FILE, "r", encoding="utf-8") as f:
    faq_data = json.load(f)

faq_questions = [item["question"] for item in faq_data]

vectorizer = TfidfVectorizer(stop_words="english")
faq_matrix = vectorizer.fit_transform(faq_questions)

SIMILARITY_THRESHOLD = 0.35


def save_escalated_query(user_query, best_similarity, best_faq_id=None):
    """Append escalated query to a JSON file so admin can review later."""
    record = {
        "user_query": user_query,
        "timestamp": datetime.now().isoformat(timespec="seconds"),
        "similarity": float(best_similarity),
        "best_faq_id": best_faq_id
    }

    if os.path.exists(ESCALATION_FILE):
        with open(ESCALATION_FILE, "r", encoding="utf-8") as f:
            try:
                data = json.load(f)
            except json.JSONDecodeError:
                data = []
    else:
        data = []

    data.append(record)

    with open(ESCALATION_FILE, "w", encoding="utf-8") as f:
        json.dump(data, f, indent=2, ensure_ascii=False)
```

```python
def get_best_answer(user_query: str):
    """Return best matched FAQ answer and similarity score."""
    if not user_query or not user_query.strip():
        return None, 0.0, None

    user_vec = vectorizer.transform([user_query])

    similarities = cosine_similarity(user_vec, faq_matrix)[0]

    best_idx = similarities.argmax()
    best_score = similarities[best_idx]

    best_faq = faq_data[best_idx]
    return best_faq["answer"], best_score, best_faq["id"]
```


```
{
  "id": 39,
  "category": "placements",
  "question": "Are off-campus placements supported?",
  "answer": "The placement cell may assist with off-campus opportunities when available."
},
{
  "id": 40,
  "category": "placements",
  "question": "Does the university conduct mock interviews?",
  "answer": "Yes, mock interviews are conducted to prepare students for real placements."
},
{
  "id": 41,
  "category": "events",
  "question": "What types of events are conducted on campus?",
  "answer": "The campus hosts cultural, technical, academic, and social events throughout the year."
},
{
  "id": 42,
  "category": "events",
  "question": "Is there an annual cultural fest?",
  "answer": "Yes, the university organizes a yearly cultural festival for students."
},
{
  "id": 43,
  "category": "events",
  "question": "Can students participate in inter-college competitions?",
  "answer": "Yes, students are encouraged to participate in inter-college events and competitions."
},
{
```

Screenshots:





Conclusion:

The Smart Enquiry Chatbot successfully demonstrates the use of AI and NLP techniques to automate enquiry services in a university environment. It provides instant responses to common queries, reduces administrative workload, and enhances user satisfaction.

This system presents a practical and cost-effective solution that can be further improved and deployed across educational institutions to modernize their support systems.

Overall, this project helped in understanding real-time web application development, natural language processing basics, and user experience design. With further enhancements, this chatbot can become a fully intelligent digital assistant for students and university visitors.