**AI-CHATBOT**

Gadhiya Shivani

Computer science and engineering

Department, PIT Parul University

**210305105512@paruluniversity.ac.in**

Prof. Arpita Vaidya

Assistant Professor,

Computer science and engineering

Department PIT,

Parul University

**Abstract**

In today's digital era, colleges and universities receive numerous queries regarding admissions, fees, courses, transportation, and scholarships. Manually handling these queries is time-consuming, inconsistent, and limited to working hours. To address these challenges, we developed an AI-powered college chatbot integrated with the Gemini API and MongoDB. The chatbot provides instant, AI-generated responses based on stored data while preserving information accuracy. Users can interact via buttons for specific queries, ensuring structured and efficient communication. This system enhances accessibility, reduces workload, and ensures 24/7 availability, improving student engagement and institutional efficiency.

**Keywords:** AI Chatbot, College Query System, Automated Response, MongoDB, Gemini API

**Introduction of Project**

Colleges and universities receive a high volume of student queries daily, ranging from admissions, fees, courses, transportation, and scholarships. Traditionally, these queries are handled manually by administrative staff, leadingtoseveral inefficiencies, including time constraints, inconsistent responses, and limited availability. The increasing number of students and their expectations for instant responses have highlighted the need for a more efficient and automated solution.

To overcome these challenges, this project introduces an AI-powered college chatbot that leverages Gemini API and MongoDB to provide real-time, accurate, and structured responses. The chatbot is designed to:

Offer instant replies to frequently asked student queries. Provide accurate and consistent information retrieved from a well-structured database. Operate 24/7, ensuring accessibility beyond working hours. Reduce the workload of administrative staff, allowing them to focus on critical tasks.

Enable a structured and user-friendly interaction using buttons for specific queries.

 By integrating AI capabilities with a reliable database, the chatbot ensures seamless communication between students and the institution, ultimately transforming the way college inquiries are managed.

**Literature Review**

AI chatbots have revolutionized student support services by providing instant and automated responses to common queries. These chatbots reduce manual intervention, allowing administrative staff to focus on complex issues. Studies indicate that institutions leveraging chatbots experience a 40% reduction in query resolution time compared to traditional methods. Natural Language Processing (NLP) plays a crucial role in enhancing chatbot efficiency. highlights the importance of machine learning models in understanding and responding to user queries. The study suggests that chatbots equipped with NLP techniques, such as intent recognition and entity extraction, can handle both structured (button-based) and unstructured (text-based) queries effectively. The integration of AI APIs like Gemini API has significantly enhanced chatbot capabilities. Research suggests that pre-trained AI models help generate human-like responses while ensuring information accuracy. A comparative analysis of chatbots using rule-based vs. AI-driven models found that AI-integrated chatbots improve user satisfaction by 60% due to dynamic response generation.

**Methodology**

The development of the AI-powered college chatbot involves multiple phases, including data collection, preprocessing, AI model integration, and deployment. This methodology outlines the steps taken to create an efficient chatbot that provides instant and structured responses to student queries.

1. Data Collection

The chatbot requires structured and accurate data to respond effectively. The dataset includes:

College Information: Admission procedures, fee structures, course details, scholarship opportunities, transportation facilities, and exam schedules.

User Queries: Common questions asked by students regarding the institution.

Database Storage: Data is stored in MongoDB, a NoSQL database, ensuring scalability and quick retrieval.

2. Data Preprocessing

Before integrating the chatbot with AI, the data undergoes preprocessing:

Data Cleaning: Removing duplicate, outdated, or incorrect information:

Data Structuring: Organizing the information into a well-structured format for easy retrieval.

Categorization: Grouping data into predefined categories such as fees, admissions, courses, etc., to improve response efficiency.

3. AI Model Integration

The chatbot is integrated with the Gemini API, which enhances response generation using AI-driven natural language processing (NLP). The process includes:

User Input Processing: Understanding user queries and mapping them to predefined categories.

Response Generation: AI retrieves the most relevant information from the MongoDB database.

Accuracy Check: Ensuring AI-generated responses align with stored data, preventing misinformation.

4. User Interface Development

The chatbot interface is designed using HTML, CSS, and JavaScript to provide an intuitive experience:

Button-based Navigation: Users select predefined queries (admission, fees, courses, etc.) for structured interactions.

Text Input Support: Allowing free-text queries for dynamic question handling.

Error Handling Mechanism: If a user asks something outside the predefined scope, the chatbot provides an appropriate error message.

5. Deployment & Testing

Backend Development: Using Python and Flask to connect the frontend with the AI model and database.

Integration Testing: Ensuring smooth interaction between the chatbot, database, and AI model.

User Testing: Conducting real-time tests with students to refine the chatbot's responses and improve user experience.

6. Performance Evaluation & Future Enhancements

Response Accuracy: Evaluating the correctness of chatbot replies.

User Engagement: Measuring how effectively students interact with the chatbot.

Scalability: Planning for future improvements such as multilingual support, voice-based queries, and additional features.

**Steps of Algorithm**

**Step 1: Initialization**

1. Start the chatbot system.

Load the MongoDB database containing college-related information (admission, fees, courses, scholarships, transportation, etc.).

Establish a connection with the GeminiAPI for AI-driven responses.

**Step 2: User Input Processing**

Display the chatbot interface with predefined query buttons (Admission, Fees, Courses, etc.).

**Step 3: Query Analysis & Classification**

If the user selects a button-based query, map it to the corresponding category in MongoDB and retrieve stored information.

If the user enters free text, process it as follows:

Use Natural Language Processing (NLP) to understand the intent. Check if the query matches existing database keywords. If matched, retrieve the relevant response from MongoDB. If not matched, forward the query to Gemini API for AI-based response generation.

**Step 4: Response Generation**

Retrieve the best-matching answer from: MongoDB (if available). AI-generated response (if no exact match in the database). Format the response and ensure it maintains accuracy.

**Step 5: Display Output**

Display the chatbot’s response in the chat window. If the response is AI-generated, include a disclaimer (e.g., "AI-generated response; verify with official sources if needed").

**Step 6: Error Handling & Feedback**

If the chatbot cannot find a relevant answer: Display an error message: *"Sorry, I couldn’t find relevant information. Please contact the college administration. "*Log the query for future training and improvements.

**Step 7: Continuous Learning & System Enhancement**

Store user queries and feedback to refine the chatbot’s accuracy. Regularly update the MongoDB database with new college-related information. Monitor chatbot performance and enhance features based on user engagement.

**Step 8: Exit**

If the user chooses to exit, terminate the session. Log session details for analysis and improvement.

**Flowchart of the System**



**Screenshots**







Conclusion

This research highlights the importance of sentiment analysis in understanding user feedback and improving app quality. By applying machine learning techniques to Play Store reviews, we provide actionable insights for developers and users. Future work could explore deep learning models for enhanced sentiment classification and incorporate multilingual support to analyse reviews in different languages. This study emphasizes the significance of AI-powered chatbots in automating college-related queries, improving student engagement, and enhancing institutional efficiency. By integrating MongoDB for structured data storage and the Gemini API for AI-generated responses, the chatbot ensures quick, accurate, and 24/7 assistance to students and faculty.

The structured button-based interaction minimizes miscommunication, while NLP-based text processing allows for flexibility in addressing free-form queries. The chatbot reduces the administrative workload, making information retrieval efficient and seamless.