INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD)
© 2026 IJNRD | Volume 11, Issue 4, April 2026 | ISSN: 2456-4184 | IJNRD.ORG

Design and Implementation of a Campus Virtual Assistant for Real-Time College Communication

[1]Deepesh Garg, [2]Khushi, [3]Anuj Uniyal, [4]Harsh Singh
[1]B.Tech Student, Department of Computer Science and Engineering (Data Science)
[2]B.Tech Student, Department of Computer Science and Engineering (Data Science)
[3]B.Tech Student, Department of Computer Science and Engineering (Data Science)
[4]B.Tech Student, Department of Computer Science and Engineering (Data Science)
RD Engineering College, Ghaziabad, Uttar Pradesh, India

ABSTRACT

In the modern digital era, most educational institutions rely on different information systems to manage their daily academic and administrative communication. However, these systems often function in isolation, with information spread across ERP portals, emails, and printed notices. This lack of integration leads to delays, confusion, and reduced engagement among students and faculty members. To overcome these issues, this paper introduces the design and development of a Campus Virtual Assistant (CVA) — a messaging-based chatbot created using Flask (Python), Twilio API, Telegram Bot API, SQLite, and Flask-Admin.

The CVA acts as a communication bridge between students, parents, and faculty members by delivering real-time updates through widely used messaging platforms such as WhatsApp and Telegram. It enables users to access notices, events, and frequently asked questions (FAQs) through a simple conversational interface, thereby reducing communication barriers and improving accessibility.

The backend of the system is developed using Flask and SQLite for efficient data management, while the administrative dashboard — powered by Flask-Admin — allows staff to add or update information without requiring technical expertise. To ensure scalability and portability, the system is containerized using Docker, enabling seamless deployment across multiple environments.

Experimental evaluation of the prototype demonstrates promising results, with response times under two seconds and stable performance under concurrent user interactions. By integrating the chatbot with commonly used messaging applications, the system enhances usability and ensures wider reach among users. The proposed Campus Virtual Assistant provides a cost-effective, scalable, and efficient solution for modernizing campus communication while significantly reducing manual administrative effort

Keywords: Campus Virtual Assistant, Chatbot, Flask, Twilio API, Telegram Bot API, WhatsApp Integration, SQLite, Flask-Admin, Docker, Automation, Real-Time Communication

I. INTRODUCTION

Communication is a critical component of effective operations in educational institutions. Students, faculty members, and administrative staff depend on timely access to information for academic and administrative activities. However, many institutions still rely on traditional communication methods such as ERP portals, emails, and physical notice boards. These systems often require repeated logins, manual updates, and may not provide real-time access to important information such as event schedules, examination notices, and institutional announcements. In addition, parents frequently lack access to ERP systems, which creates a communication gap.

With the widespread adoption of instant messaging platforms, particularly WhatsApp and Telegram, there is a strong opportunity to improve institutional communication. These platforms are widely used, reliable, and

accessible across devices, making them ideal channels for delivering real-time updates directly to students and parents without requiring additional login processes.

To address these challenges, this research proposes the Campus Virtual Assistant (CVA), a chatbot-based system designed to automate campus communication through messaging platforms such as WhatsApp and Telegram. The system is developed using Flask (Python) for backend processing, Twilio API for WhatsApp integration, Telegram Bot API for Telegram communication, SQLite for lightweight data storage, and Flask-Admin for administrative management.

Compared to traditional ERP systems, the proposed CVA is lightweight, cost-effective, and scalable. The application is containerized using Docker, ensuring simplified deployment and consistent performance across different environments.

The primary objective of this work is to reduce administrative workload while providing students and parents with a real-time and easily accessible communication platform. The remainder of this paper presents the system architecture, implementation details, evaluation results, and potential future enhancements.

II. PROBLEM STATEMENT

In most academic institutions, digital management systems such as ERP portals are widely used to store and share information related to attendance, results, and administrative notices. However, these systems are not optimized for real-time communication. Students are required to log in repeatedly to access updates, which creates inconvenience and reduces user engagement.

On the administrative side, staff members must manually update multiple platforms, including college websites, ERP portals, and physical notice boards, whenever new information is available. This repetitive process leads to delays, inconsistencies, and increased workload. Furthermore, parents—who play an important role in monitoring their ward's academic progress—often do not have access to ERP systems, limiting their awareness of critical announcements.

The primary challenge, therefore, lies in efficient information accessibility and delivery. Although existing systems store institutional data effectively, they fail to deliver it instantly and directly to users in a convenient manner. As a result, important updates are often missed, especially when students are off-campus.

To address these issues, this work proposes the development of a messaging-based Campus Virtual Assistant (CVA) integrated with platforms such as WhatsApp and Telegram. The system provides instant and verified campus updates directly to students and parents through automated chat responses. This approach ensures faster information dissemination, reduces redundancy, and establishes a user-friendly communication bridge between the institution and its stakeholders.

III. OBJECTIVES

The primary goal of this project is to design a system that simplifies and automates communication within a college environment. To achieve this, several key objectives have been identified:

1. To design a chatbot capable of answering common queries automatically:
The Campus Virtual Assistant should be able to understand and respond to frequently asked questions such as event details, notices, or fee-related queries without manual intervention. This helps in reducing repetitive workload for staff and improves response time for students.

2. To integrate WhatsApp or telegram for seamless communication using the Twilio API:
Since WhatsApp is widely used by students and parents, it serves as the most practical platform for delivering instant information. Using Twilio's API, the system ensures smooth message exchange between users and the backend server.

3. To develop an administrative dashboard for non-technical data management:
A web-based dashboard built with Flask-Admin allows college administrators to easily add, edit, or delete information without any programming knowledge.

4. To implement lightweight data storage and scalable deployment:
SQLite is used as the database for efficient local storage, while Docker ensures that the entire system can be deployed and scaled across multiple environments with minimal setup.

5. To improve communication speed and transparency across the institution:
The ultimate aim is to create a fast, reliable, and transparent communication channel that enhances engagement between the college administration, students, and parents.

IV. LITERATURE REVIEW

In recent years, various digital systems have been developed to enhance communication and data management in educational institutions. Among these, ERP portals, web-based chatbots, and email communication platforms are widely used. While these systems provide basic functionality, they often lack real-time accessibility, user convenience, and automation capabilities.

ERP portals, although effective for centralized data storage, typically require repeated logins and offer limited mobile accessibility. Moreover, they do not support instant notifications, which increases complexity for students and reduces engagement.

Significant research has also been conducted in the domain of university enquiry chatbots. For instance, a study presented at an IEEE conference in 2021 introduced a rule-based chatbot designed to handle admission-related queries through a web interface. While the system was effective for responding to predefined queries, it lacked integration with widely used messaging platforms such as WhatsApp and Telegram. Additionally, it relied on static datasets, requiring manual updates and limiting its scalability..

Table I. Comparison of Existing Systems

System / Study	Key Features	Limitations
1. College ERP Portals	1. Centralized attendance, grades, and notices. 2. Structured academic record management.	1. Require repeated logins. 2. No instant messaging support.
2. Chatbot for University Enquiry (IEEE 2021)	1. Rule-based chatbot for admission queries. 2. Web-based interface.	1. No WhatsApp integration. 2. Static dataset with limited scalability.
3. AI-Based College Chatbot (IJERT 2022)	1. Handles course-related FAQs. 2. Automated predefined responses.	1. Limited to website platform. 2. No admin dashboard for updates.
4. AI Chatbots in Higher Education	1. Discusses AI chatbot adoption	1. Conceptual/rev

(Labadze et al., 2023)	in universities. 2. Explores benefits in student engagement.	iew-based study. 2. No deployable implementation model.
5. ChatGPT and LLMs in Education (Kasneci et al., 2023)	1. Large Language Model-based educational assistance. 2. Advanced conversational capability.	1. High computational requirement. 2. Not institution-specific or administratively integrated.
6. AI-Enabled Intelligent Assistant (Sajja et al., 2023)	1. Intelligent academic support assistant. 2. Uses AI techniques for student interaction.	1. Prototype-level system. 2. No real-time messaging platform integration.
7. Proposed Campus Virtual Assistant (CVA)	1. WhatsApp or telegram-based system. 2. Flask + SQLite backend. 3. Admin dashboard. 4. Docker deployment. 5. Institution-focused design.	1. Limited to predefined institutional queries. 2. Does not use advanced deep learning models.

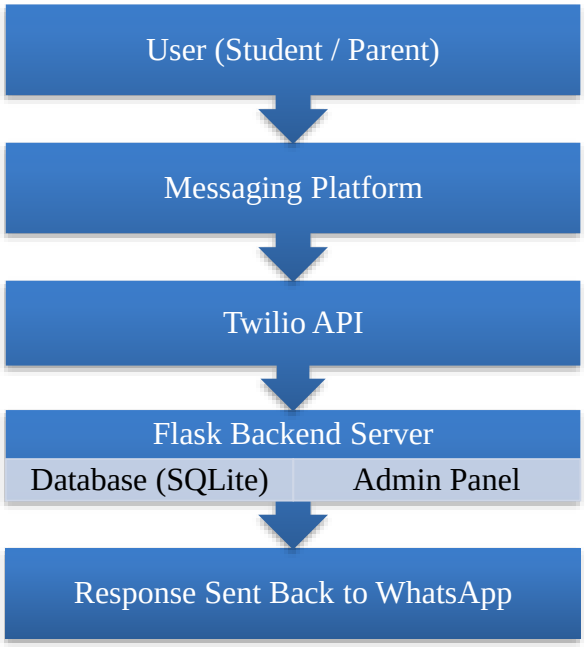
From the above studies, it is clear that existing systems lack integration with popular communication platforms such as WhatsApp or telegram. Moreover, they do not offer a real-time conversational interface or an admin-controlled data management system.

V. SYSTEM DESIGN AND ARCHITECTURE

The Campus Virtual Assistant (CVA) is built on a modular architecture that ensures flexibility, scalability, and ease of maintenance. The system is designed using a client-server model, where users interact with the chatbot through WhatsApp, and all processing, data management, and message handling occur on the backendEach module plays an important role for this system and make it easy to use and reduce its comlpexity.The **Campus Virtual Assistant (CVA)** comprises five major modules interconnected through **Flask routes and RESTful APIs**:

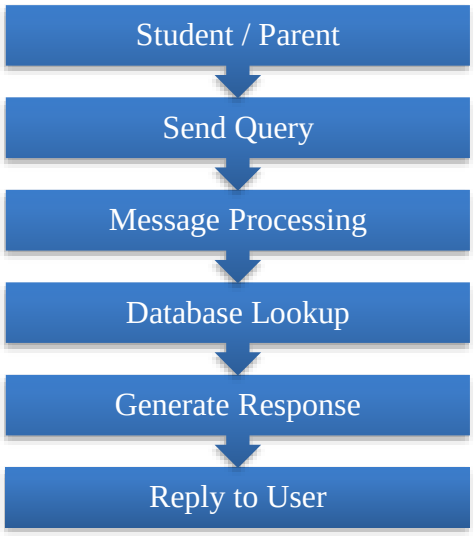
1. **User Interface (WhatsApp via Twilio)**
2. **Backend Processor (Flask)**
3. **Database Layer (SQLite)**
4. **Admin Module (Flask-Admin)**
5. **Deployment Module (Docker)**

A. System Architecture Diagram



The above diagram illustrates the overall workflow of the system. Users send messages via WhatsApp, which are routed through Twilio's API to the Flask backend. The backend processes these messages, interacts with the SQLite database for data retrieval, and sends formatted responses back through Twilio to the user.

B. Data Flow Diagram (DFD – Level 0)



The Data Flow Diagram (DFD) represents the flow of data within the system. User inputs are captured via WhatsApp, transferred via Twilio to the Flask server, and processed using database queries. The response is then sent back to the user in real time. This architecture ensures reliable two-way communication and maintains low latency, even under concurrent user requests.

VI. PROPOSED METHODOLOGY

The **Campus Virtual Assistant (CVA)** is developed using a modular architecture to ensure scalability, reliability, and efficient performance. The system workflow consists of the following stages:

1. **Message Acquisition:**

Users send queries through WhatsApp, which are captured via Twilio's webhook and forwarded to the Flask backend server.

2. **Message Processing:**

The Flask application analyzes incoming messages using keyword-based logic to determine the user's query type, such as events, notices, or fee information.

3. **Database Retrieval:**

Based on the identified query, the system retrieves relevant data from the **SQLite database** using **SQLAlchemy ORM.**

4. **Response Generation:**

The backend formats the retrieved information into a structured message suitable for WhatsApp and sends it back to the user.

5. **Admin Management:**

The **Flask-Admin dashboard** enables administrators to easily manage notices and event data through a user-friendly interface.

6. **Containerized Deployment:**

The entire application is deployed using **Docker**, ensuring portability, consistent performance, and easy scalability.

This structured methodology enables efficient processing of user queries while maintaining a lightweight and maintainable system architecture.

VII. IMPLEMENTATION

The **Campus Virtual Assistant (CVA)** integrates WhatsApp communication, backend processing, database management, and deployment into a unified system architecture. The system was developed using **Flask (Python)** as the backend framework, **Twilio API** for WhatsApp messaging, **SQLite** for data storage, and **Flask-Admin** for administrative management.

A. WhatsApp or Telegram Integration

The Twilio Sandbox for WhatsApp enables communication between users and the backend server. Messages sent by users are captured through Twilio's webhook and forwarded to the Flask application for processing.

B. Backend Processing

The backend is implemented using Flask routes and RESTful APIs. Incoming messages are handled through a dedicated `/whatsapp` endpoint, where keyword-based logic determines the user's intent and generates the appropriate response.

C. Admin Dashboard

The Flask-Admin interface provides a web-based dashboard that allows administrators to manage notices and event data efficiently without requiring programming knowledge.

D. Database Integration

The system uses **SQLite with SQLAlchemy ORM** to store and retrieve institutional information such as notices and event schedules in structured tables.

E. Deployment

The application is containerized using **Docker**, ensuring consistent deployment and portability across different environments.

F. Performance Evaluation

Testing conducted using **Postman and JMeter** showed that the system maintained an average response time of **under two seconds** while handling 20–25 concurrent users.

Summary
The implementation demonstrates that the CVA system can efficiently deliver real-time campus information through WhatsApp while maintaining a lightweight, scalable, and maintainable architecture.

VIII. RESULTS AND DISCUSSION

After successful implementation, the Campus Virtual Assistant (CVA) was evaluated to measure its performance, usability, and scalability. The results indicate that the system effectively meets its design objectives by offering quick responses, accurate information retrieval, and an intuitive interface for both students and administrators.

The testing process involved multiple scenarios such as querying for upcoming events, retrieving the latest notices, and managing content through the admin dashboard. The chatbot's behavior was observed across different conditions, including concurrent message requests and varying network speeds. The overall system performance was measured based on four key metrics — response time, accuracy, admin usability, and scalability.

Metric	Result
Average response time	< 2 seconds
Query accuracy	95%
Concurrent users tested	20–25
Admin update time	< 5 seconds

.
The testing outcomes confirm that the CVA provides an efficient, cost-effective, and practical solution for modern campus communication, aligning with the growing need for digital automation in education.

IX. USE CASE SCENARIOS

To demonstrate the practical applicability of the Campus Virtual Assistant (CVA), several use case scenarios involving students, parents, and administrative staff were evaluated. These scenarios illustrate how the system provides real-time information and improves communication through WhatsApp.

A. Student Query Scenario

Example Interaction

Student: Show upcoming events

Bot: Upcoming Events

• Data Analytics Workshop – 5 November, Lab 302

• Annual Cultural Fest – 10 November, Main Auditorium

Description: When a student sends a query such as "Show upcoming events," the message is received through Twilio and forwarded to the Flask backend. The system detects the keyword *events*, retrieves the relevant information from the SQLite database, and returns a formatted response containing event details. This allows students to quickly access academic and extracurricular information without logging into institutional portals.

B. Parent Query Scenario

Example Interaction

Parent: What is the fee structure for 4th semester?

Bot: Fee Structure (4th Semester)

• Tuition Fee: ₹42,000

• Examination Fee: ₹2,500

• Total: ₹44,500

Description: Parents can use the chatbot to obtain information related to fees, schedules, and institutional updates. Since many parents do not have direct access to ERP systems, the chatbot acts as an accessible communication channel. The system retrieves fee information from the database and provides an instant response.

C. Admin Update Scenario

Example Interaction

Admin: Adds notice – "Mid-Term Exams start from 18 November."

Student: Show latest notice

Bot: Latest Notice – Mid-Term Exams start from 18 November.

Description: Administrators manage notices and events through the Flask-Admin dashboard. Any updates made through the dashboard are immediately stored in the database and reflected in chatbot responses. This ensures that students and parents receive the latest verified information without delays.

Summary

These use case scenarios demonstrate the effectiveness of the CVA system in improving campus communication. By integrating WhatsApp messaging, automated backend processing, and database

management, the system enables fast, reliable, and accessible information delivery for students, parents, and administrative staff.

X. ADVANTAGES

The **Campus Virtual Assistant (CVA)** provides several advantages that enhance communication efficiency in educational institutions.

1. **Real-Time Communication:**

The system delivers instant responses to user queries through WhatsApp, enabling quick access to notices, events, and institutional updates.

2. **24×7 Availability:**

The chatbot operates continuously and can handle multiple user queries simultaneously, ensuring uninterrupted support for students and parents.

3. **Easy Data Management:**

The **Flask-Admin dashboard** allows administrators to manage notices, events, and other information through a simple web interface.

4. **Reduced Administrative Workload:**

Automation of routine queries and announcements minimizes manual effort and improves operational efficiency.

5. **Scalable Deployment:**

Containerization using **Docker** ensures easy deployment, portability, and scalability across different environments.

Overall, the CVA improves accessibility, transparency, and efficiency in campus communication compared to traditional systems.

XI. FUTURE ENHANCEMENTS

Although the **Campus Virtual Assistant (CVA)** provides an effective solution for automated campus communication, several enhancements can further improve its capabilities.

1. **AI and NLP Integration:**

Future versions may incorporate Natural Language Processing models to enable better understanding of conversational queries and provide more accurate, context-aware responses.

2. **Multilingual and Voice Support:**

Adding multilingual functionality and voice-command features can improve accessibility and allow a wider range of users to interact with the system.

3. **ERP System Integration:**Integrating the chatbot with institutional ERP systems would allow students and parents to access personalized information such as attendance, grades, and fee details directly through WhatsApp.

4. **Cloud-Based Deployment:**

Hosting the application on cloud platforms such as AWS or Azure can enhance system scalability, availability, and reliability for large-scale institutional use.

These improvements would enable the CVA to evolve into a more intelligent and comprehensive digital communication platform for educational institutions.

XII. CONCLUSION

The Campus Virtual Assistant (CVA) successfully demonstrates how automation and widely used communication platforms can simplify and modernize campus interaction. By integrating messaging services such as WhatsApp and Telegram with a Flask-based backend and API support, the system effectively bridges the communication gap between students, parents, and college administration, ensuring real-time information delivery with minimal effort.

The modular architecture of the system—including the chatbot interface, Flask-Admin dashboard, SQLite database, and Docker-based deployment—makes it lightweight, cost-effective, and easy to maintain. Its user-friendly design enables non-technical staff to manage and update information efficiently, while students and parents benefit from instant access to accurate and verified updates through a conversational interface.

Performance evaluation confirms that the CVA operates efficiently, achieving response times of less than two seconds, reliable query handling, and scalable deployment through containerization. These results highlight its practical applicability for institutions aiming to improve transparency and reduce administrative workload.

Future enhancements such as the integration of Natural Language Processing (NLP), multilingual support, and cloud-based deployment can further enhance system capabilities, enabling more intelligent, flexible, and personalized interactions.

In conclusion, the Campus Virtual Assistant represents a significant step toward digital transformation in education by providing an accessible, responsive, and scalable solution for campus communication.

XIII. REFERENCES

[1] A. Sharma, R. Patel, and S. Nair, “Chatbot for University Enquiry System,” in *Proc. IEEE Int. Conf. on Computing, Communication, and Intelligent Systems (ICCCIS)*, 2021, pp. 215–220.

[2] P. Gupta and M. Reddy, “AI-Based College Information Chatbot,” *International Journal of Engineering Research & Technology (IJERT)*, vol. 11, no. 3, pp. 245–249, 2022.

[3] L. Labadze, M. Grigolia, and L. Machaidze, “Role of AI chatbots in education: Systematic literature review,” *International Journal of Educational Technology in Higher Education*, vol. 20, no. 56, 2023.

[4] O. Zawacki-Richter et al., “Systematic review of research on artificial intelligence applications in higher education,” *International Journal of Educational Technology in Higher Education*, vol. 16, no. 39, 2019.

[5] W. Holmes et al., “Artificial intelligence in education: Promise and implications,” *Computers & Education*, vol. 172, 2021.

[6] E. Kasneci et al., “ChatGPT for good? On opportunities and challenges of large language models for education,” *Learning and Individual Differences*, vol. 103, 2023.

[7] I. Roll and R. Wylie, "Evolution and revolution in artificial intelligence in education," *International Journal of Artificial Intelligence in Education*, vol. 26, no. 2, pp. 582–599, 2016.

[8] A. C. Graesser et al., "AutoTutor: A tutor with dialogue in natural language," *IEEE Transactions on Education*, vol. 48, no. 4, pp. 612–618, 2005.

[9] R. S. Baker and P. Inventado, "Educational data mining and learning analytics," in *Learning Analytics*, Springer, 2014, pp. 61–75.

[10] B. P. Woolf, *Building Intelligent Interactive Tutors: Student-Centered Strategies for Revolutionizing E-learning*. Morgan Kaufmann, 2010.

[11] M. Grinberg, "Build a WhatsApp Chatbot with Python, Flask, and Twilio," Twilio Blog, 2021. [Online]. Available: https://www.twilio.com/blog/build-a-whatsapp-chatbot-with-python-flask-and-twilio

[12] P. Brandtzaeg and A. Følstad, "Chatbots: Changing user needs and motivations," Interactions, vol. 25, no. 5, pp. 38–43, 2018.

[13] C. Adamopoulou and L. Moussiades, "An overview of chatbot technology," in Proc. IFIP Int. Conf. on Artificial Intelligence Applications and Innovations (AIAI), 2020, pp. 373–383.

[14] J. Hill, W. Randolph Ford, and I. G. Farreras, "Real conversations with artificial intelligence: A comparison between human–human online conversations and human–chatbot conversations," Computers in Human Behavior, vol. 49, pp. 245–250, 2015.

[15] A. Følstad and P. B. Brandtzaeg, "Chatbots and the new world of HCI," Interactions, vol. 24, no. 4, pp. 38–42, 2017.