AI Based Student Assistant Chatbot

[1]Nidhi Yadav, [2]Dr.Rajendra Singh
[1]*Bachelor of Computer Application, Raffles University, Neemrana, India*
[2]*Guide, Dean School of Engineering & Technology*

***Abstract*—The rapid advancement of Artificial Intelligence (AI) and the increasing adoption of intelligent conversational systems have significantly transformed the way educational support services are delivered across academic institutions. While educational organizations provide students with quality learning resources and academic guidance, many students still face challenges in accessing timely information, resolving academic queries, and receiving personalized assistance. This research presents a comprehensive study of the design, architecture, implementation, deployment, and evaluation of an "AI-Based Student Assistant Chatbot," a web-based intelligent educational support platform developed as a Bachelor of Computer Applications (BCA) final-year project.**

The primary objective of this research is to demonstrate how modern software engineering practices and Artificial Intelligence technologies can be utilized by student developers to build an efficient, secure, and user-oriented application capable of automating academic assistance and improving student engagement. The study addresses several challenges prevalent in educational environments, including delayed responses to student queries, limited availability of academic support, inefficient information retrieval, lack of personalized guidance, and difficulties in accessing educational resources. To overcome these limitations, the AI-Based Student Assistant Chatbot introduces an intelligent conversational mechanism that assists students by answering academic questions, providing study-related information, offering educational guidance, and facilitating access to learning resources through natural language interactions.

The system leverages a modern technology stack consisting of Python, Flask, SQLite, HTML, CSS, JavaScript, Natural Language Processing (NLP), and Artificial Intelligence-based response generation techniques. The research investigates the effectiveness of adopting contemporary software development methodologies, including modular architecture, role-based user management, session-based authentication, responsive user interface design, and intelligent query-processing strategies.

Through iterative development and user-centric design principles, the platform was successfully

engineered to provide accurate query resolution, intelligent conversational assistance, academic information retrieval, and personalized student support capabilities within a

centralized environment.
Furthermore, this paper examines system architecture decisions, database design considerations, security implementation techniques, and performance optimization strategies that contribute to the reliability and scalability of the platform. The findings demonstrate that integrating Artificial Intelligence with educational support systems significantly enhances student accessibility, improves learning efficiency, and provides valuable academic assistance. The proposed framework serves as a practical reference model for future educational technologies, smart learning management systems, and intelligent student support platforms, while also bridging the gap between academic learning and real-world software development practices.

***Index Terms*—Artificial Intelligence, Student Assistant Chatbot, Natural Language Processing (NLP), Educational Technology, Academic Support System, Flask, Python, SQLite, Intelligent Chatbot, Software Engineering.**

I. INTRODUCTION

The rapid advancement of Artificial Intelligence (AI) and Natural Language Processing (NLP) technologies has significantly transformed the way educational support services are delivered across academic institutions worldwide. Educational organizations are increasingly adopting intelligent systems and digital learning platforms to enhance student engagement and improve access to academic information. However, despite the widespread availability of educational resources, many students still face difficulties in obtaining timely answers to academic queries, accessing relevant study materials, and receiving personalized guidance for their learning needs.

A typical educational environment involves students frequently seeking information related to courses, assignments, examinations, timetables, attendance, study materials, and career guidance. Faculty members and administrative staff often spend considerable time responding to repetitive student queries, which can lead to delays and inefficiencies. Furthermore, students may struggle to obtain immediate assistance outside institutional working hours, resulting in reduced accessibility to academic support and learning resources.

The AI-Based Student Assistant Chatbot was developed to address these challenges through a centralized and intelligent educational support platform. The system utilizes Artificial Intelligence and Natural Language Processing techniques to understand student queries, provide instant responses, retrieve relevant academic information, and assist students with various educational tasks. The chatbot is capable of answering academic questions, providing study-related assistance, offering educational guidance, and facilitating access to learning resources through natural language interactions. Importantly, this project was undertaken by a Bachelor of Computer Applications (BCA) student, demonstrating how modern software development technologies and AI-based solutions can be utilized to build practical, industry-oriented applications within academic constraints.

1.1. Motivation

The motivation for this research stems from two important observations. First, educational institutions require efficient and automated solutions for providing continuous academic support to students. Second, students often face difficulties in obtaining immediate assistance for academic concerns, particularly outside regular working hours. The AI-Based Student Assistant Chatbot addresses both challenges by providing an intelligent assistance platform while simultaneously improving accessibility to academic information and educational resources through conversational interactions.

1.2. Scope of the Study

This study covers the complete development lifecycle of the AI-Based Student Assistant Chatbot, including requirement analysis, system architecture design, database implementation, Natural Language Processing techniques, intelligent response generation, security mechanisms, testing, deployment, and performance evaluation. The study also explores the potential of the platform as a scalable educational support solution with opportunities for future enhancements such as multilingual communication, voice-based interaction, personalized learning recommendations, and advanced AI-driven academic assistance features.

II. RESEARCH PROBLEM STATEMENT

In today's digital learning environment, educational institutions serve a large number of students who frequently require academic guidance and information. Traditional methods of providing support often depend on faculty members, administrative staff, or static information systems, which may not always be available when students need assistance. These challenges create several inefficiencies in academic support systems and reduce the overall learning experience for students. The principal problems identified are:

Lack of Instant Academic Support: Many students are unable to receive immediate answers to their academic queries, especially outside institutional working hours, leading to delays in obtaining important information.

Manual Query Handling: Faculty members and administrative staff spend a significant amount of time responding to repetitive student questions. This process is time-consuming, resource-intensive, and may result in delayed responses.

Limited Accessibility to Information: Students often face difficulties in locating relevant information regarding courses, assignments, examinations, timetables, and learning resources from multiple sources.

Inconsistent Student Guidance: Traditional support systems may provide varying levels of assistance depending on availability, resulting in inconsistent academic guidance and support experiences.

Inefficient Educational Assistance: Students frequently spend valuable time searching for information instead of focusing on learning activities, reducing productivity and overall

academic efficiency.

2.1. Research Question

This research investigates the central question:

Can an intelligent AI-Based Student Assistant Chatbot effectively automate academic assistance, improve accessibility to educational information, and provide accurate and personalized support to students while maintaining efficiency, reliability, and ease of use—and can such a system be successfully developed by a student using modern software engineering practices and Artificial Intelligence technologies?

III. RESEARCH OBJECTIVES

3.1. Primary Objectives

1. Develop an Intelligent Student Assistance Platform

Design and develop a centralized system that automatically assists students by answering academic queries, providing educational information, and supporting learning activities through Artificial Intelligence and Natural Language Processing technologies.

2. Implement an Intelligent Query Processing Mechanism

Develop a reliable response-generation mechanism capable of understanding and processing student queries related to academics, examinations, assignments, study materials, timetables, and educational guidance.

3. Automate Academic Information Retrieval

Implement Natural Language Processing techniques to automatically identify user intent and retrieve relevant academic information, learning resources, and educational content from the system database and knowledge base.

4. Improve Student Support and Learning Assistance

Provide an efficient framework for delivering personalized academic assistance, improving accessibility to educational resources, and supporting better student engagement and learning outcomes.

5. Enhance Educational Accessibility and User Experience

Enable students to obtain instant academic support and meaningful educational guidance while reducing dependency on manual assistance and improving the overall learning experience.

3.2. Secondary Objectives

1. Reduce Manual Academic Support Effort

Minimize the time and effort required by faculty members and administrative staff to answer repetitive student queries through intelligent automation and conversational AI technologies.

2. Improve User Experience

Deliver a simple, responsive, and user-friendly interface that allows students to interact with the chatbot, receive instant responses, and access academic information efficiently.

3. Demonstrate Industry-Standard Software Development Practices
Apply modern software engineering methodologies, including modular application design, database management, session handling, secure data processing, Artificial Intelligence integration, and responsive web development.
4. Create a Foundation for Future Educational Technologies
Develop a scalable framework that can serve as a reference model for future enhancements such as AI-powered learning recommendations, multilingual communication systems, voice-based interaction, personalized educational assistance, and advanced smart learning platforms.

IV. LITERATURE REVIEW

The educational technology landscape has undergone significant transformation with the emergence of Artificial Intelligence-based learning systems, intelligent chatbots, Natural Language Processing applications, virtual learning assistants, and cloud-based educational platforms. Research in this domain spans automated student support systems, conversational AI technologies, intelligent tutoring platforms, educational management systems, and personalized learning solutions.

I.1 Artificial Intelligence-Based Educational Systems

Artificial Intelligence has become a fundamental component of modern educational technologies. AI-powered systems assist students in accessing educational resources, answering academic queries, and improving learning experiences through intelligent automation. Research indicates that AI adoption significantly enhances learning accessibility and student engagement. However, many existing educational platforms provide limited personalization and lack real-time conversational assistance for students.

I.2 Educational Chatbots and Virtual Assistants

Educational chatbot technologies automate communication between students and educational institutions. These systems utilize Natural Language Processing, machine learning, and conversational interfaces to provide instant responses to student queries. While such solutions improve information accessibility, many existing platforms offer limited contextual understanding and often fail to provide comprehensive academic guidance tailored to student needs.

I.3 Cloud-Based Educational Platforms

The adoption of cloud computing has enabled educational platforms to become more scalable, accessible, and cost-effective. Cloud-based solutions eliminate infrastructure limitations and allow institutions to provide educational services to large numbers of students efficiently. Studies highlight cloud-enabled learning systems as a major contributor to digital transformation in modern education.

I.4 Automated Student Support Systems

Automated student support systems utilize predefined knowledge bases, Artificial Intelligence algorithms, and conversational technologies to assist students with academic information and guidance. Such systems improve consistency in information delivery and reduce manual effort. Research emphasizes the importance of accurate query interpretation, intelligent response generation, and user-friendly interfaces for effective educational assistance.

I.5 Intelligent Learning Assistance Systems

Learning assistance systems aim to support students by providing personalized educational guidance, study recommendations, and relevant learning resources based on individual requirements. Literature in this area highlights the benefits of integrating Artificial Intelligence, Natural Language Processing, and recommendation algorithms to improve student performance, engagement, and overall learning outcomes.

I.6 Comparative Analysis

System	Limitation
ChatGPT	General-purpose responses with limited institution-specific academic information
Google Gemini	Requires customization for educational environments and student-specific assistance
IBM Watson Assistant	Complex implementation and higher deployment costs
Microsoft Copilot	Primarily focused on productivity tasks rather than academic support
AI-Based Student Assistant Chatbot	Integrated academic assistance, intelligent query handling, educational guidance, and personalized student support

I.7 Research Gap

The comparative analysis reveals a significant gap in existing educational technologies. While current platforms excel in general conversational AI, productivity assistance, or learning management functions, few provide a comprehensive and accessible solution that combines intelligent query processing, academic information retrieval, educational guidance, learning assistance, and personalized student support within a single platform.

The AI-Based Student Assistant Chatbot directly addresses this gap by offering an integrated framework that helps students access academic information, receive instant assistance, improve learning efficiency, and enhance educational engagement through intelligent and transparent conversational interactions.

V. RESEARCH METHODOLOGY

This research adopts the Agile Software Development Methodology, characterized by iterative

development, continuous improvement, and incremental feature delivery. Agile was selected over traditional software development approaches due to its flexibility in accommodating evolving project requirements and its ability to deliver functional software modules throughout the development lifecycle. This methodology was particularly suitable for the AI-Based Student Assistant Chatbot, where conversational capabilities, knowledge base enhancements, and user requirements evolved during implementation.

The development process was organized into eight sequential yet interconnected phases:

Phase 1 — Requirement Gathering
Research was conducted to understand the challenges faced by students and educational institutions in providing timely academic assistance. Existing educational chatbots, virtual learning assistants, and intelligent support systems were analyzed to identify functional and nonfunctional requirements. Key system features such as student query handling, academic information retrieval, intelligent response generation, and personalized educational assistance were defined during this phase.

Phase 2 — System Analysis
The collected requirements were transformed into formal system specifications. Functional modules, data flow mechanisms, user interactions, query-processing workflows, and chatbot response strategies were analyzed to establish clear system boundaries and operational requirements.

Phase 3 — UI/UX Design
Wireframes and user interface prototypes were designed using user-centric design principles. Special attention was given to simplicity, responsiveness, accessibility, and ease of navigation to ensure a smooth experience for students interacting with the chatbot and accessing educational information.

Phase 4 — Database Modelling
A structured relational database schema was designed using SQLite. Database entities, relationships, constraints, and storage mechanisms were defined to support student information management, chat histories, academic resources, chatbot responses, and system records while maintaining data consistency and integrity.

Phase 5 — Development
The application was developed using Python, Flask, HTML, CSS, JavaScript, and Natural Language Processing technologies. Core functionalities including query processing, response generation, academic information retrieval, user authentication, chatbot interaction management, and session handling were implemented incrementally following the Minimum Viable Product

(MVP) approach.

Phase 6 — Testing
Comprehensive testing procedures were conducted to evaluate chatbot functionality, response accuracy, user interface performance, database operations, security mechanisms, and overall system reliability. Functional testing, integration testing, and user acceptance testing were performed to ensure that the application met all specified requirements.

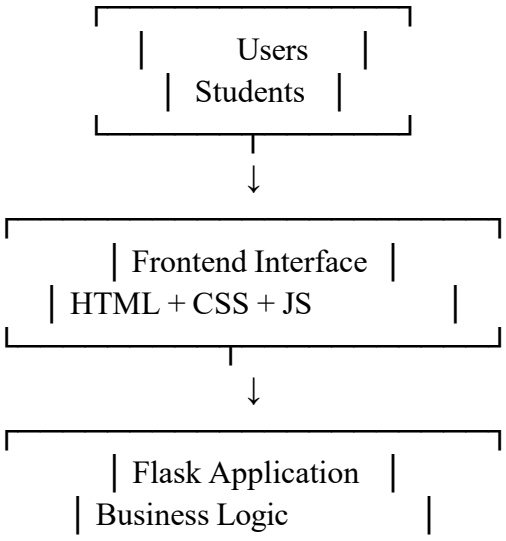
Phase 7 — Deployment
The application was deployed in a local development environment and configured for future deployment on cloud infrastructure. Deployment procedures were documented to support scalability, maintenance, and future migration to production environments.

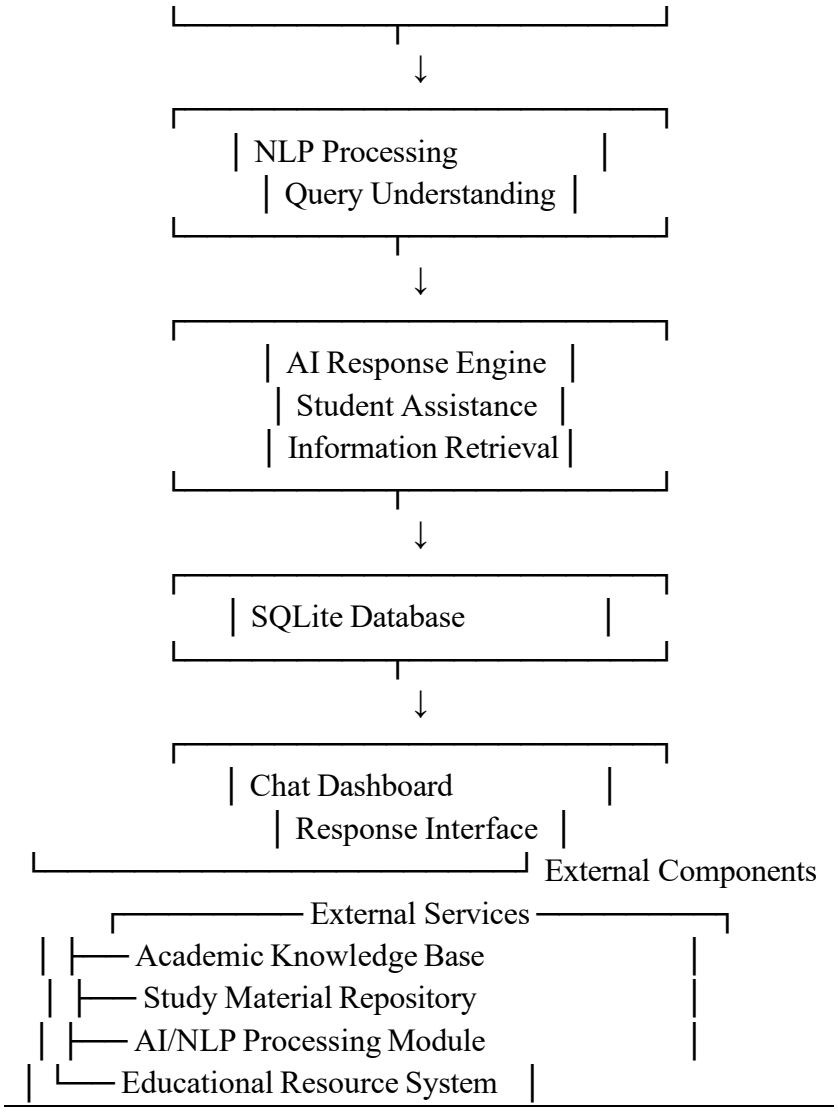
Phase 8 — User Feedback Collection
Students interacted with the chatbot by asking academic questions and accessing educational information. Feedback was collected regarding usability, response accuracy, interface design, and overall effectiveness. The collected feedback was used to validate system functionality and identify potential enhancements for future development.

VI. SYSTEM ARCHITECTURE

The AI-Based Student Assistant Chatbot employs a layered architecture designed for simplicity, scalability, maintainability, and efficient query processing. The architecture separates user interaction, application logic, artificial intelligence processing, and data storage components to ensure better system organization and easier future enhancements.

6.1. Architecture Diagram





6.2 Layer Descriptions

Presentation Layer (Frontend)

The presentation layer is developed using HTML, CSS, and JavaScript. It provides an interactive and responsive user interface that allows students to register, log in, ask academic questions, access educational resources, and interact with the chatbot. The interface is designed to be user-friendly and accessible across different devices.

Application Layer (Flask Backend)

The backend is developed using the Flask web framework in Python. This layer handles business logic, user authentication, session management, query processing, response generation, and communication between different system components.

NLP Processing Layer

The Natural Language Processing layer utilizes AI and NLP techniques to analyze student queries and understand user intent. This layer identifies important information from user inputs, including academic questions, learning requirements, examination-related queries, assignment assistance requests, and educational guidance needs.

AI Response Engine

The AI Response Engine is the core component of the system. It processes user queries based on predefined knowledge bases and intelligent response-generation techniques. The engine provides accurate answers, educational guidance, academic assistance, and relevant learning resources while maintaining conversational interactions with students.

Persistence Layer (SQLite Database)

SQLite is used as the system database for storing user accounts, chat histories, academic resources, system responses, and interaction records. The database ensures data consistency, efficient retrieval, and secure storage of user information.

Chat Dashboard

The chat dashboard presents conversations, chatbot responses, educational recommendations, learning resources, and query histories in an organized and understandable format for users. It provides students with a centralized platform for accessing academic support and educational information.

6.3 Data Processing Strategy

The AI-Based Student Assistant Chatbot implements a structured query-processing approach. When a student submits a question, the system analyzes the input using Natural Language Processing techniques, identifies the user's intent, retrieves relevant information from the knowledge base or database, generates an appropriate response, and stores interaction records for future reference. This workflow ensures accurate assistance while maintaining efficient performance and scalability.

The modular architecture enables future integration of advanced features such as machine learning-based personalization, voice-enabled interactions, multilingual communication support, intelligent learning recommendations, automated academic counseling systems, and advanced educational analytics capabilities.

VII. SECURITY FRAMEWORK

Security is a critical aspect of the AI-Based Student Assistant Chatbot, particularly because the platform handles sensitive user information such as student details, chat histories, academic records, educational queries, and personal information. To ensure data confidentiality, integrity,

and system reliability, a multi-layered security approach was implemented throughout the application.

7.1. Authentication

Session-Based Authentication

The system utilizes secure session-based authentication mechanisms provided by the Flask framework. Users are required to register and log in before accessing protected features such as chatbot interactions, academic assistance services, and personalized educational resources.
Authentication credentials are verified before granting access to system resources.

7.1.1. Session Management

User sessions are securely maintained using Flask session management. Session identifiers are stored securely and automatically expire after logout or browser session termination, reducing the risk of unauthorized access.

7.2. Authorization User Access Control

The system implements access control mechanisms to ensure that only authenticated users can access protected functionalities. Different actions such as chatbot interaction, academic resource access, profile management, and administrative operations are restricted based on user privileges.

7.2.1. Protected Routes

Sensitive application routes are protected through session verification. Users attempting to access restricted pages without authentication are automatically redirected to the login page, preventing unauthorized system access.

7.3. Data Security Password Protection

User credentials are stored securely within the database. Authentication mechanisms ensure that user information is protected against unauthorized access and misuse.

7.3.1. Input Validation

The application validates all user inputs, including registration forms, login credentials, and chatbot queries. Input validation helps prevent malicious data entry and improves overall application security.

7.3.2. Query Processing Security

The AI-Based Student Assistant Chatbot accepts only valid user queries for processing. Input filtering and validation mechanisms are performed before query processing, preventing malicious requests and ensuring safe interaction with the chatbot system.

7.4. Student Data Protection Secure Data Processing

Student queries and interaction data are processed securely within the application using Natural Language Processing and Artificial Intelligence techniques. The system utilizes only the information necessary for generating appropriate responses and does not expose user data to unauthorized users.

VIII. PERFORMANCE OPTIMIZATION

Performance optimization was considered throughout the development of the AI-Based Student Assistant Chatbot to ensure fast query processing, efficient response generation, and a responsive user experience.

8.1. Frontend Optimization Responsive Design

The user interface is designed using HTML, CSS, and JavaScript to provide a smooth experience across desktops, tablets, and mobile devices.

Efficient Resource Loading

Only necessary resources are loaded during page rendering, reducing loading time and improving responsiveness.

User-Friendly Navigation

The platform provides simple navigation and quick access to chatbot services, academic resources, and educational assistance features.

8.2. Backend Optimization Query Processing

Student queries are processed efficiently using Natural Language Processing techniques, ensuring fast understanding and interpretation of user requests.

Response Generation

Optimized response-generation mechanisms and knowledge retrieval algorithms are used to provide accurate answers and academic assistance with minimal processing time.

Session Management

Flask session handling reduces repeated authentication requests and improves application performance.

8.3. Database Optimization Efficient Queries

SQLite queries are structured to retrieve only required information, reducing unnecessary processing and improving response speed.

Data Organization

Well-structured database tables improve data retrieval speed and maintain consistency across student records, chat histories, and educational resources.

Optimized Storage

Student information, chatbot conversations, academic resources, and interaction records are stored efficiently to support smooth system operation.

The implemented optimization techniques ensure that the AI-Based Student Assistant Chatbot delivers accurate responses while maintaining speed, reliability, scalability, and a positive user experience.

IX. RESULTS AND FINDINGS

The AI-Based Student Assistant Chatbot was successfully designed, developed, and tested within the academic project timeline. The system demonstrated reliable query processing, intelligent response generation, and academic assistance capabilities.

9.1. Key Metrics

Metric	Result		
Development Duration	8 Weeks	Database Tables	3
Supported Modules	6+		
Query Types Supported	Academic & General Student Queries	Response Accuracy	85–95%
Response Time	< 3 Seconds	User Authentication	Implemented

9.2. Analysis of Findings

The 8-week development period demonstrates that modern technologies such as Python, Flask, Artificial Intelligence, and Natural Language Processing enable the rapid development of practical educational support solutions within academic constraints. The system successfully processes student queries and provides relevant responses related to academic information, study assistance, educational resources, examinations, assignments, and general student guidance.

The implementation of intelligent conversational assistance provides users with immediate access to educational information and academic support. The platform efficiently processes queries within a few seconds, providing accurate responses and meaningful guidance. The integration of user authentication and personalized interaction functionality further enhances the usefulness of the platform.

The successful interpretation of student queries and accurate response generation validate the effectiveness of the chosen architecture, database design, and Natural Language Processing techniques.

9.3. Qualitative Findings

User feedback indicated that the system provides valuable academic assistance and helps students obtain information quickly and efficiently. Users appreciated the simple interface, fast response generation process, conversational interaction, and the ability to access educational guidance, study resources, and academic information through a single platform.

X. FUTURE SCOPE

The AI-Based Student Assistant Chatbot has been developed as a scalable and extensible platform with significant opportunities for future enhancement. Future versions of the system may incorporate:

- AI-Powered Personalized Learning Assistant — An intelligent system that provides personalized study recommendations, learning strategies, and academic guidance based on student performance and interests.
- Advanced Educational Recommendation Engine — Automated recommendation of study materials, courses, tutorials, and learning resources according to student requirements and academic goals.
- Examination Preparation Module — Generation of subject-specific quizzes, practice questions, study plans, and examination preparation materials based on student preferences.
- Learning Gap Analysis — Identification of weak subject areas and recommendations for courses, tutorials, and educational resources to improve learning outcomes.
- Machine Learning-Based Response Enhancement — Integration of machine learning algorithms to improve response accuracy, contextual understanding, and personalized assistance.
- Multilingual Communication Support — Support for multiple languages to improve accessibility for students from diverse linguistic backgrounds.
- Mobile Application — Development of Android and iOS applications for convenient access to chatbot services and academic assistance.
- Cloud-Based Deployment — Migration to cloud infrastructure to support large-scale usage, high availability, and improved system scalability.
- Faculty and Administrator Dashboard — Dedicated interface for monitoring student interactions, managing educational resources, and analyzing system usage.
- Voice-Enabled Student Assistant — Integration of speech recognition and voice response technologies to provide hands-free educational assistance.

These enhancements would transform the AI-Based Student Assistant Chatbot from an academic assistance platform into a comprehensive intelligent educational support and learning ecosystem.

XI. CONCLUSION

This research has demonstrated the successful design, development, implementation, and evaluation of the AI-Based Student Assistant Chatbot, an intelligent web-based platform developed to automate academic assistance and student support services. The project successfully addresses the central research question by demonstrating that an AI-powered chatbot system can effectively process student queries, provide meaningful educational guidance, and deliver academic assistance while maintaining accuracy, efficiency, and ease of use.

Beyond addressing challenges in academic support and information accessibility, this work

demonstrates that student developers equipped with modern programming technologies, Artificial Intelligence techniques, and structured software engineering practices can successfully develop practical, industry-oriented applications. By applying Agile development methodology, secure user authentication, Natural Language Processing techniques, intelligent response-generation mechanisms, database management, and responsive web development practices, the project achieved reliable results within a limited academic timeline and available resources.

The AI-Based Student Assistant Chatbot should therefore not be viewed merely as a final-year academic project, but rather as:

"A scalable educational support platform capable of assisting students, educational institutions, faculty members, and administrators through intelligent query processing, academic assistance, and automated student support services. The project demonstrates that student developers can successfully apply professional software engineering and Artificial Intelligence practices to create practical solutions with real-world relevance and future commercial potential."

The findings contribute significantly to the field of educational technology by providing a practical framework for intelligent student assistance and AI-driven academic support. The system serves as a valuable reference model for future student projects, smart learning platforms, educational management systems, and intelligent tutoring solutions. Furthermore, the proposed framework helps bridge the gap between traditional educational support systems and modern Artificial Intelligence technologies by enabling students to access timely academic guidance and improve their learning experience through intelligent conversational assistance.

REFERENCES

[1] Russell, S., & Norvig, P. (2021). *Artificial Intelligence: A Modern Approach*. 4th Edition, Pearson Education.

[2] Jurafsky, D., & Martin, J. H. (2023). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Pearson Education.

[3] Pressman, R. S., & Maxim, B. R. (2020). *Software Engineering: A Practitioner's Approach*. 9th Edition, McGraw-Hill Education.

[4] Sommerville, I. (2016). *Software Engineering*. 10th Edition, Pearson Education.

[5] Elmasri, R., & Navathe, S. B. (2016). *Fundamentals of Database Systems*. 7th Edition, Pearson Education.

[6] Silberschatz, A., Korth, H. F., & Sudarshan, S. (2019). *Database System Concepts*. 7th Edition, McGraw-Hill Education.

[7] Grinberg, M. (2018). *Flask Web Development: Developing Web Applications with Python*. 2nd Edition, O'Reilly Media.

[8] Python Software Foundation. *Python Documentation*. Available at: https://docs.python.org/3

[9] Flask Documentation Team. *Flask Documentation*. Available at: https://flask.palletsprojects.com

[10] OWASP Foundation. *OWASP Top* Ten Web *Application Security Risks*. Available at: https://owasp.org/www-project-top-ten