JIKO (Jurnal Informatika dan Komputer) Accredited KEMDIKTISAINTEK, No.0173/C3/DT.05.00/2025
Vol. 9, No. 2, August 2026, pp. 93-99 p-ISSN: 2614-8897
DOI: 10.33387/jiko.v9i2.11579 e-ISSN: 2656-1948

TELEGRAM-BASED CHATBOT DESIGN FOR ACADEMIC SERVICE AUTOMATION AT FASILKOM UEU USING A LOW-CODE PLATFORM

Gabriela Marry Christiana Simanjuntak[1*], Dewi Setiowati[2], Binastya Anggara Sekti[3], Sawali Wahyu[4]

[1] Information System, Faculty of Computer Science, Universitas Esa Unggul, Jakarta, 11510, Indonesia
[2,3,4] Informatics Engineering, Faculty of Computer Science, Universitas Esa Unggul, Jakarta, 11510, Indonesia

Email: [*1]gabrielamarrych@student.esaunggul.ac.id, [2]dewi.setiowati@esaunggul.ac.id, [3]anggara@esaunggul.ac.id, [4]sawaliwahyu@esaunggul.ac.id

(Received: 05 March, Revised: 22 May 2026, Accepted: 27 June 2026)

Abstract

Administrative services at the Faculty of Computer Science, Universitas Esa Unggul, face persistent challenges due to high volumes of repetitive inquiries and limited operational hours, which may reduce service efficiency and responsiveness. This study aims to design, implement, and evaluate FasilkomAssist, a 24/7 Telegram-based chatbot developed using the ADDIE framework and a Low-Code/No-Code (LCNC) approach via the n8n workflow automation platform. The system integrates Google Gemini as a Large Language Model (LLM) with Google Sheets, Google Drive, Google Calendar, and Telegram to automate the dissemination of Thesis information and administrative procedures. Evaluation was conducted through Black-box testing, the System Usability Scale (SUS), and User Acceptance Testing (UAT) involving 30 respondents. Results indicate a 96% response accuracy across 17 valid testing scenarios, with an average response time of 4–6 seconds. The SUS score of 70.75 places the system within the "Acceptable" range (Grade C), indicating satisfactory usability with opportunities for optimization. Furthermore, UAT conducted with Faculty Administration Staff yielded a score of 84.67%, confirming functional adequacy and user acceptance. These findings demonstrate that an LCNC-based chatbot can improve efficiency, accessibility, and service quality in academic administrative environments without requiring advanced programming expertise.

Keywords: *Chatbot, Automation, Low-Code No-Code, n8n, Google Gemini, ADDIE*

**Corresponding Author: Gabriela Marry Christiana Simanjuntak*

1. INTRODUCTION

Advances in Low-Code/No-Code (LCNC) technology have accelerated digital transformation by enabling application development and process automation without requiring advanced programming skills. In this study, the n8n platform was used to integrate Google Gemini with Google services (Google Sheets, Google Drive, and Google Calendar) as well as Telegram. The selection of Gemini over other AI models was based on its strong compatibility with the Google ecosystem and its relatively cost-efficient API, thereby supporting optimal workflow orchestration within n8n.

In higher education, information technology plays an important role in improving academic services, particularly in the thesis process, which involves complex administrative procedures. Observations at the Faculty of Computer Science, Universitas Esa Unggul identified several challenges, including repetitive inquiries, low engagement with lengthy guidelines, and limited administrative service hours. These issues lead to inconsistent information dissemination and increased administrative workload, highlighting the need for a fast, consistent, and 24/7 accessible information service.

Based on a review of previous studies, several differences were identified between prior research and the present research. Previous research developed a

Telegram chatbot using the Wit.ai platform, focusing on providing information related to campus agendas, faculty contacts, and scholarships [1]. Meanwhile, other studies have utilized the n8n platform; however, its implementation was limited to website-based

systems, with a focus on improving operational performance and financial efficiency in Islamic cooperatives at the Bank Indonesia Representative Office of Aceh Province [2]. So far, no study has been found that utilizes the n8n platform to support academic administrative services, particularly for critical processes such as undergraduate thesis administration, through a Telegram chatbot. Therefore, this study offers novelty in terms of platform utilization, implementation, and application context.

Chatbots have emerged as a relevant solution to address these challenges. Previous research [3] indicates that studies on the use of chatbots in education increased significantly during the 2021–2025 period and demonstrated positive impacts across various learning contexts. Higher education institutions were identified as the educational level with the highest chatbot adoption rate.

This study aims to design an automated, integrated Thesis information service, construct a dynamic knowledge base derived from primary faculty data, and adapt to technological advancements [4], and evaluate the system's effectiveness in terms of response time, accuracy, and user satisfaction. Additionally, the chatbot can contribute to a more supportive administrative experience for students and reflects the implementation of human–computer interaction in academic digital services [5].

2. RESEARCH METHOD

This study adopts the ADDIE system development approach (Analysis, Design, Development, Implementation, and Evaluation) [6], can be seen in Figure 1.

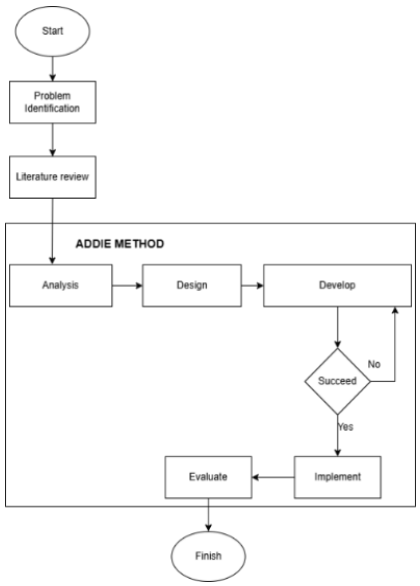


Figure 1. Stages of Research Method

The development stages are summarized as follows:

a) Analysis: This stage involves data collection, identification of functional and non-functional chatbot requirements, service objectives, technical constraints, and selection of software components (Telegram, Google Sheets, Google Calendar, Google Drive, Gemini, and Sumopod).
b) Design: This phase focuses on designing system workflows and interactions, developing the chatbot knowledge base, and defining the chatbot interface.
c) Development: The design is implemented as workflows in n8n, integrating message handling, conversational logic, and external APIs, followed by functional testing to ensure response accuracy and data validity.
d) Implementation: The developed system is deployed to a group of end users for initial real-time testing and feedback collection to identify technical issues.
e) Evaluation: System performance is assessed based on response accuracy, responsiveness, functionality, and usability, with evaluation results used for continuous system improvement.

2.1 Analysis Functional and Non-Functional

Functional requirements define the core functions that the system must provide to meet user needs and serve as a reference throughout the development process. Table 1 presents the functional requirements of the developed Telegram bot.

Table 1. Functional requirements

Role	Requirement	Function
Computer Science Faculty Students	Timeline & Requirements	Receive information regarding the Proposal Seminar & Thesis timeline & requirements.
	Proposing Supervisors	Know how to propose Thesis Supervisors.
	Title Change	Receive information on how to change the Thesis title.
	Certificate Upload Tutorial	Receive tutorials regarding uploading certificates / SKPI Certificates in SIAKAD.
	BNSP Registration	Know the registration flow for BNSP certificates & how to validate.
	Zoom Background & PPT Template	Students receive a link containing the Proposal Seminar – Thesis files
	File of Thesis Template & Academic Calendar	Students receive the Thesis writing guideline file & Academic calendar.
	Lecturer Schedule	Students know the lecturer's schedule.
Admin	CRUD Database	Faculty Admin can perform CRUD (Create, Read, Update, Delete) operations on the database.
	Monitor Workflow	Faculty Admin can monitor the workflow.

Non-functional requirements address system quality attributes such as performance, scalability, security, and usability to ensure stable and reliable operation, significantly affecting user experience and system dependability [7].

Table 2. Non-functional requirements

No	Requirement	Function
1.	Performance	The chatbot can respond to user requests with minimal time delay.
2.	Ease of use	The chatbot is easy to use so that it can be operated without the need for complex learning.

2.2 Proposed Business Process

The current process depends on manual admin responses in a Telegram group, causing unanswered queries due to the lack of automation and a structured knowledge base. The proposed process employs an n8n-integrated Telegram bot to automate intent handling and LLM-based responses, with unresolved queries escalated for knowledge base updates and performance tracking.

2.3 Use Case Diagram

A Use Case Diagram is a UML diagram that illustrates the interaction between actors and the system, as well as the main functions provided, helping to define the system's scope and behavior from the user's perspective [8]. The system involves students and Faculty Administration Staff, where students interact with the Telegram bot, and admins manage FAQs, schedules, and files through CRUD operations.

2.4 Activity Diagram

An Activity Diagram is a type of UML diagram used to illustrate the flow of activities or processes within a system being designed. It shows how a process starts, the sequence of activities performed, possible decision points, parallel processes, and how the process ends. [9]. When a student submits a query, the system classifies the intent and entity to identify whether the request is general, schedule-related, or file-based. The chatbot then validates through Google Drive, Google Calendar, or Google Sheets, and returns the requested information. If data is unavailable, the request is automatically escalated to Faculty Administration Staff for updates.

2.5 Sequence Diagram

A Sequence Diagram illustrates the sequence of interactions between objects at runtime, showing how the system responds to user actions within a use case [10]. Just like activity diagrams, sequence diagrams are classified into 3 schemes, namely: general (non-schedule & non-file related), schedule-related queries, and file-related queries.

2.6 Class Diagram

A Class Diagram is a UML diagram that shows the system's structure through classes, their attributes and methods, and the relationships between them [11]. The class diagram illustrates the relationships among components in the n8n workflow, where students interact through a Telegram Bot connected to n8n as the process orchestrator and API integrator. n8n coordinates AI Gemini and accesses Google Sheets, Google Drive, and Google Calendar with support from Sumopod (CaaS), while Faculty Administration Staff manage and monitor the data.

2.7 Component Diagram

A Component Diagram shows how software components are structured and integrated within a system. The component diagram illustrates the chatbot architecture integrated through APIs, consisting of the Telegram Bot, Container Services, and the n8n workflow. Within n8n, AI Gemini processes natural language queries and accesses the knowledge base, including databases, schedules, and files, before delivering responses to users via the Telegram Bot.

2.8 Telegram Bot Creation

The Telegram Bot account was created via BotFather to obtain the bot name and API token. After the API token is obtained, the bot is activated by opening the Telegram link and using the /start command, the API token is used to connect the web system with the bot. Further configurations, including updating the bot name, bio, and description, are managed through BotFather.

3. RESULT AND DISCUSSION

3.1 Workflow Implementation

The n8n workflow was designed to manage a Telegram-based chatbot for administrative services related to proposal seminars and thesis. The process begins with a Telegram Trigger, followed by intent analysis using the Gemini AI model. The parsed results are then classified through rule-based logic into file requests, lecturer schedules, student defense schedules, or FAQ services. Relevant data are retrieved from Google Drive, Google Calendar, or Google Sheets, then formatted and delivered to users. If the requested information is unavailable, the system notifies the Faculty Administration Staff for data updates.

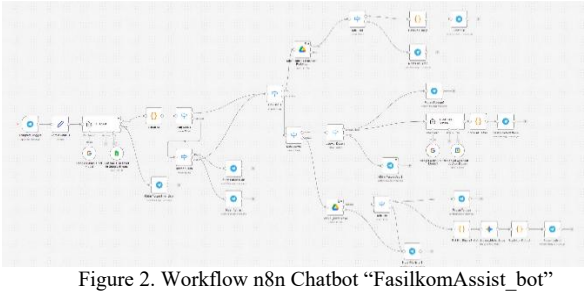


Figure 2. Workflow n8n Chatbot "FasilkomAssist_bot"

3. 1 Sumopod Subscription

Sumopod is utilized as a Container as a Service (CaaS) platform to deploy n8n. After registration and billing top-up, the n8n Plus service was selected and deployed. Once activated, the service is accessible through the admin console using the domain sumopod.my.id.

3.2 Credential & API: Telegram

Telegram functions as both the user interface and workflow trigger. The bot token obtained from BotFather is configured in the Telegram Trigger credentials within n8n, and a successful connection is confirmed through system notification.

3.3 Credential & API: AI Gemini

The Gemini API Key was generated via Google AI Studio after creating a Google Cloud project. Due to Free Tier limitations, Tier 1 was selected. The API Key was configured in the Gemini credential node in n8n, with Gemini 2.5 Flash used as the selected model.

3.4. Credential & API: Google Drive, Sheets, and Calendar

Google Drive, Sheets, and Calendar APIs were enabled through Google Cloud Console. OAuth Consent Screen configuration was completed using an External audience type. The generated Client ID and Client Secret were then integrated into the respective n8n credentials to enable secure access to files, schedules, and FAQ data.

3.5 Testing Scope

Based on the evaluation scope and indicators, this study employs black-box testing and usability evaluation. Black-box testing measures the chatbot's response time and answer accuracy based on response duration and relevance to user intent and correct data, without analyzing internal processes. Usability evaluation is conducted using the System Usability Scale (SUS) and User Acceptance Testing (UAT) by Faculty Administration Staff to assess usability, ease of use, and user acceptance. Therefore, the evaluation covers both technical performance and user acceptance aspects.

Evaluation Aspect	Measured Indicator	Measurement Method	Expected Result
Response Time	Average response time of the chatbot in responding to user requests.	**Average Response Time =** Total Response Time ÷ Number of Requests	Average response time ≤ 20 seconds
Answer Accuracy	Accuracy and relevance of the chatbot's responses based on primary data and user intent.	**Answer Accuracy =** (Number of Correct Answers ÷ Total Conversations) × 100%	Accuracy rate ≥ 75%
SUS Questionnaire by Respondents	Consists of 10 statements measured using a 5-point Likert Scale ranging from "Strongly Disagree" to "Strongly Agree."	**SUS Score =** ((R1−1) + (5−R2) + (R3−1) + (5−R4) + (R5−1) + (5−R6) + (R7−1) + (5−R8) + (R9−1) + (5−R10)) × 2.5	Average SUS score ≥ 65
UAT Questionnaire by Faculty Admin	Consists of 10 statements evaluating the chatbot's functionality using a 5-point Likert Scale from "Very Poor" to "Very Good."	**Success Percentage =** (Total Actual Score ÷ Total Maximum Score) × 100%	Average success rate ≥ 80%

Figure 3. Testing Scope

3.6. Black Box Testing

Black-box testing was conducted on the FasilkomAssist_bot to verify that all system functions operate according to defined requirements and specifications [12]. The testing consisted of 10 core scenarios representing the chatbot's core features and services. The results indicate that all scenarios were valid, confirming that the system consistently produces correct outputs for given user inputs.

Table 3. Blackbox Testing

No	Test Scenario	Test Results
1	Inquiring about the timeline for the Thesis	Valid
2	Inquiring about the requirements for the Thesis	Valid
3	Inquiring about the procedure for proposing Thesis Supervisors	Valid
4	Inquiring about the procedure for revising the Thesis title	Valid
5	Inquiring about the procedure for uploading participation certificates to SIAKAD	Valid
6	Inquiring about administrative assets	Valid
7	Inquiring about the Thesis Writing Guidelines and templates	Valid
8	Inquiring about a lecturer's schedule (faculty availability)	Valid
9	Accessing the /cakupan_bot menu	Valid
10	Handling unrecognized queries	Valid

3.7. Response Time Testing by the Author

Testing was conducted to determine the average response time interval of FasilkomAssist_bot to 17 test scenarios run by the author. It was divided into four time intervals: 1-3s, 4-6s, 7-9s, and 10-12s. According to [13], descriptive statistics are used to organize and present data in a structured and informative manner. In this study, the median time interval is calculated as (upper limit + lower limit) / 2.

Interval	Median (Xi)	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Q11	Q12	Q13	Q14	Q15	Q16	Q17	Total Response
1-3s	2																		0
4-6s	5	1	1	1	1		1	1	1	1	1	1						1	11
7-9s	8					1							1						2
10-12s	11													1					1
Not Valid (>13s)															1	1	1		3

Figure 4. Result of Response Time Test by Author

The calculation was performed using the Average Response Time metric [14]. The following details the test results, with an average chatbot response time of 5.8 seconds.

Average Response Time = Total Response Time / Number of Requests

- Total Response Time = 82s
- Number of Successful Requests = 14
- Average Response Time 82/14 = 5.8s
- Number of Invalid Requests = 3

Of the 17 test scenario requests, 14 were successfully executed within four time interval groups. Three requests were invalid because they returned output beyond the specified time interval. Therefore, the average response time per request was 5.8s, equivalent to a 4-6s interval.

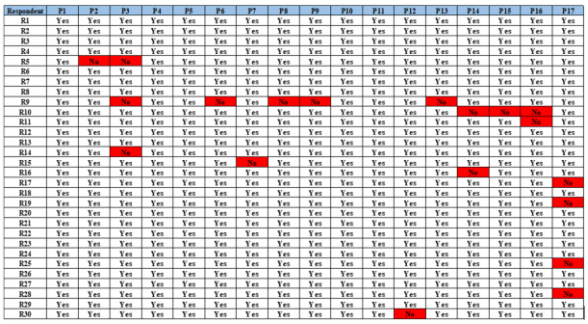
3.8 Response Time Test Results by Respondents

Based on a questionnaire consisting of 17 testing scenarios completed by 30 respondents, the chatbot

response time varied and was grouped into four time ranges. Each range was represented by its midpoint value, while 19 responses that failed to produce accurate outputs were excluded from the calculation. The total response time was obtained by multiplying the frequency of responses in each range by its midpoint value. Based on the calculation results, the average response time per request was 5.4s, equivalent to a 4-6s interval.

3.9 Results of Accuracy Testing of Answers by Respondents

The following figure presents the questionnaire results filled out by respondents regarding the bot's response accuracy for the 17 defined testing scenarios.



Respondent	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15	P16	P17
R1	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R2	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R3	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R4	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R5	Yes	No	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R6	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R7	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R8	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R9	Yes	Yes	No	Yes	Yes	No	Yes	No	No	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes
R10	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	Yes
R11	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
R12	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R13	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R14	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R15	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R16	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes
R17	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
R18	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R19	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
R20	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R21	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R22	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R23	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R24	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R25	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
R26	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R27	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R28	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No
R29	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R30	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes

Figure 5. Result of Response Accuracy Test by Respondents

To calculate the Answer Accuracy Testing results, the following metrics were used:

Intent Accuracy = (Correct Predictions / Total Predictions) x 100%

1. Total Test Conversations: 510
2. Intent predicted correctly: 491
3. Accuracy = (491/510) x 100% = 96%

It was found that the responses provided by "FasilkomAssist_bot" were 96% accurate.

3.10 System Usability Scale (SUS) Testing by Respondents

Usability testing was conducted using the System Usability Scale (SUS) method. SUS scores were calculated by subtracting one from the responses to odd-numbered items and subtracting the responses to even-numbered items from five. All scores were summed and multiplied by 2.5 to obtain a SUS score ranging from 0 to 100, which was then averaged to evaluate the system's overall usability level. The SUS assessment indicators are presented in the following figure.

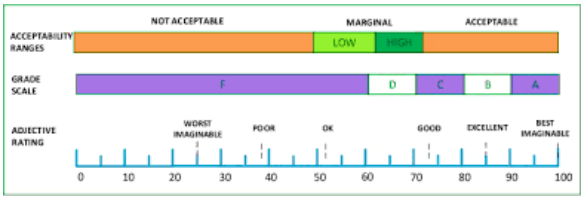


Figure 6. SUS Score Indicator

The SUS results were interpreted using three categories: *acceptability ranges*, *grade scale*, and *adjective rating*. The *acceptability ranges* include Not Acceptable (0–50.9), Marginal Low (51–62), Marginal High (63–70), and Acceptable (71–100). The *grade scale* classifies scores from Grade F to Grade A, while the *adjective rating* provides qualitative evaluations ranging from Worst Imaginable to Best Imaginable, representing the level of ease of use and user comfort of the system [15].

The usability of FasilkomAssist_bot was evaluated using the System Usability Scale (SUS), which consists of 10 statements rated on a 1–5 Likert scale [16], ranging from strongly disagree to strongly agree, with 10 statement below.

No	Statement	Point
1	I feel this system is easy to use	1-5
2	I feel this system is complicated to use	1-5
3	I feel I will use this system again	1-5
4	I need help from others or a technician to use this system	1-5
5	I feel this system runs as it should	1-5
6	I feel there are obstacles in using this system	1-5
7	I feel confident when using this system	1-5
8	I feel others will find it difficult to understand how to use this system quickly	1-5
9	I feel this system is helpful within the scope of the Computer Science Faculty Thesis	1-5
10	I feel there are many inconsistencies (not harmonious) in this system	1-5

Figure 7. SUS Questionnaire Statement List

The collected data were processed using Microsoft Excel to calculate the total scores.

Based on the System Usability Scale (SUS) score interpretation, a score of 70.75, rounded to 71, falls within the **Acceptable category**, indicating that the system is generally acceptable to users. According to the grade scale, this score is classified as **Grade C**, suggesting that the system demonstrates adequate usability while still offering opportunities for improvement. Furthermore, based on the adjective rating scale, a score of 71 is categorized as **Good**, indicating that users generally perceive the system as reasonably comfortable and easy to use.

3.11 UAT Test

User Acceptance Test (UAT) is a system testing stage carried out by end users to ensure that the application or software meets business needs and previously established specifications [17].

UAT questionnaire data is calculated using a 1-5 Likert scale, where: 1 = "Excellent"; 2 = "Good"; 3 = "Fair"; 4 = "Poor"; 5 = "Very Poor."

To generate a success percentage, each answer is inversely weighted (the lower the number, the greater the weight). Referring to research [18], the calculation of the percentage of success in UAT uses the equation:

$$Success = \frac{Total\ Actual\ Score}{Total\ Maximum\ Score} x\ 100\% \qquad (1)$$

Description:

• Total Actual Score = number of answers multiplied by their respective weights

• Total Maximum Score = number of respondents x 5 (maximum weight)

Based on the results of the User Acceptance Test (UAT) conducted by three Faculty Administration Staff on 10 indicators, the system achieved an average success rate of 84.67%, indicating that the system has met user needs and is feasible for implementation. The highest scores were for response time, level of assistance with administrative tasks, and manual workload reduction (93.33%). Meanwhile, aspects of information accuracy (73.33%) and display comfort (66.67%) still need improvement. Overall, the system is considered to be performing well with several recommendations for minor improvements.

Code	Questions	VG x 5	G x 4	F x 3	P x 2	VP x 1	Actual Score	Success Percentage
UAT1	Does the chatbot answer questions accurately?	5	8	0	0	0	13	86.67%
UAT2	Is the information provided accurate and reliable?	0	8	3	0	0	11	73.33%
UAT3	Does the chatbot provide complete and useful information?	5	8	0	0	0	13	86.67%
UAT4	Does the chatbot respond promptly?	10	4	0	0	0	14	93.33%
UAT5	Does the system operate without errors?	5	8	0	0	0	13	86.67%
UAT6	Are the menus and commands easy to understand?	0	12	0	0	0	12	80.00%
UAT7	Does the chatbot simplify administrative processes?	5	8	0	0	0	13	86.67%
UAT8	Is the system interface user-friendly?	0	8	0	2	0	10	66.67%
UAT9	Is the chatbot helpful for administrative tasks?	10	4	0	0	0	14	93.33%
UAT10	Does the chatbot reduce manual workload?	10	4	0	0	0	14	93.33%
	Number of Respondents	3	3	3	3	3		
	Average							**84.67%**

Figure 8. Result of UAT Test

4. CONCLUSION

The implementation of FasilkomAssist_bot using a low-code/no-code workflow platform (n8n), integrated with Telegram, Google Gemini, Google Calendar, Google Sheets, and Google Drive through Sumopod (CaaS), successfully provides automated thesis administrative services with 24/7 availability.

The knowledge base, built using familiar tools, enables real-time management of primary data from the Faculty of Computer Science and can be easily operated by non-technical users. Black-box testing showed that 14 of 17 requests were successfully processed within 1–12 seconds, with an average response time of 5.8 seconds (between 4s and 6s), while 3 requests were classified as invalid for exceeding the 12-second threshold.

Testing involving 30 respondents generated 510 requests, of which 491 were successfully processed, resulting in an accuracy rate of 96%. The average response time for successful requests was 5.4 seconds (between 4s and 6s).

The System Usability Scale (SUS) score of 70.75 (rounded to 71) falls within the acceptable category (Grade C; adjective rating: Good), indicating that the system is sufficiently usable and acceptable to users. However, improvements are still needed in user-friendliness, data accuracy and updates, response speed, and consistency in format and language.

User Acceptance Testing (UAT) conducted by three Faculty Administration Staff members resulted in an average success rate of 84.67%, indicating that the system meets user requirements and is feasible for implementation.

This study has several limitations. First, the system still experienced response delays in several requests, as indicated by 3 out of 17 requests in the black-box testing that exceeded the 12-second threshold. Second, although the usability evaluation showed that the system falls within the acceptable category, improvements are still required in terms of user-friendliness, response speed, data accuracy and updates, as well as consistency in format and language. Furthermore, the system was specifically developed for undergraduate thesis administrative services in the Faculty of Computer Science, and its applicability to broader academic administrative contexts has not yet been extensively evaluated. In addition, the system relies on third-party services such as Telegram, Google Gemini, Google Calendar, Google Sheets, Google Drive, and Sumopod (CaaS), which may affect system stability and performance if service disruptions or platform policy changes occur.

5. REFERENCES

[1] V. Joey Ferelestian, B. Susanto, and I. K. D. Senapartha, "Pengembangan Telegram Chatbot Informasi Mahasiswa Menggunakan Wit.ai," *Jurnal Terapan Teknologi Informasi*, vol. 7, no. 2, pp. 89–97, Oct. 2023, doi: 10.21460/jutei.2023.72.257.

[2] M. Wali, N. Nasir, and T. Iqbal, "Implementing Workflow Automation with N8N to Enhance Operational Efficiency and Performance in the Sharia Cooperative of Bank Indonesia, Aceh Province," *Journal Digital Technology Trend*, vol. 4, no. 1, pp. 36–47, Jun. 2025, doi: 10.56347/jdtt.v4i1.341.

[3] K. I. Dewi, I. M. Falasifa, and F. T. Nafasya, "A Literature Review of Chatbot Utilization in Digital Education," *Jurnal Ilmiah Sistem Informasi*, vol. 5, no. 1, pp. 306–325, Jan. 2026, doi: 10.51903/4wbpcj98.

[4] B. A. Sekti, A. P. Gusti, and N. Erzed, "Perancangan Sistem Informasi Stok Barang

berbasis Web dengan Metode FIFO," *Jurnal Teknologi Informatika dan Komputer*, vol. 10, no. 2, pp. 506–518, Sep. 2024, doi: 10.37012/jtik.v10i2.2253.

[5] Y. J. Oh, J. M. Hu, R. Zhu, J. I. Lim, and X. Zhang, "Artificial Intelligence Chatbots as Relational Agents: A Systematic Review of Human–AI Chatbot Relationships," *Int. J. Hum. Comput. Interact.*, pp. 1–25, Apr. 2026, doi: 10.1080/10447318.2026.2648799.

[6] A. khusnul Umam, E. Wijayanti, and A. A. Chamid, "Pengembangan Chatbot Pada Platform Telegram Sebagai Media Informasi Seputar Handphone," *bit-Tech*, vol. 8, no. 1, pp. 33–40, Aug. 2025, doi: 10.32877/bt.v8i1.2150.

[7] S. Wahyu and T. Beltsazar, "Development of Chatbot Integration in Personal Finance Management Applications Using Generative Pre-Trained Transformer (GPT) 4o," *Bulletin of Computer Science Research*, vol. 6, no. 1, pp. 434–444, 2025, doi: 10.47065/bulletincsr.v6i1.776.

[8] M. R. Wayahdi and F. Ruziq, "Pemodelan Sistem Penerimaan Anggota Baru dengan Unified Modeling Language (UML) (Studi Kasus: Programmer Association of Battuta)," *Jurnal Minfo Polgan*, vol. 12, no. 1, pp. 1514–1521, Aug. 2023, doi: 10.33395/jmp.v12i1.12870.

[9] S. Sandfreni, M. B. Ulum, and A. H. Azizah, "Analisis Perancangan Sistem Informasi Pusat Studi Pada Fakultas Ilmu Komputer Universitas Esa Unggul," *Sebatik*, vol. 25, no. 2, pp. 345–356, Dec. 2021, doi: 10.46984/sebatik.v25i2.1587.

[10] R. Rohmanto and T. Setiawan, "Perbandingan Efektivitas Sistem Pembelajaran Luring dan Daring Menggunakan Metode Use case dan Sequence Diagram," *INTERNAL (Information System Journal)*, vol. 5, no. 1, pp. 53–62, Jun. 2022, doi: 10.32627/internal.v5i1.506.

[11] S. Ramdany, "Penerapan UML Class Diagram dalam Perancangan Sistem Informasi Perpustakaan Berbasis Web," *Journal of Industrial and Engineering System*, vol. 5, no. 1, pp. 30–41, Jul. 2024, doi: 10.31599/2e9afp31.

[12] D. Setiowati, Q. H. Hidayah, and D. Nurmadewi, "Aplikasi Three Tier Sistem Informasi Manajemen Kepegawaian Menggunakan Model Prototype," *Bulletin of Computer Science Research*, vol. 5, no. 5, pp. 988–1001, Aug. 2025, doi: 10.47065/bulletincsr.v5i5.724.

[13] P. A. Utomo, "Sistem Informasi Smart RT/RW Menggunakan Metode Statistik Deskriptif Berbasis Website Pada Perumahan Pakal Residence Surabaya," 2021.

[14] A. khusnul Umam, E. Wijayanti, and A. A. Chamid, "Pengembangan Chatbot Pada Platform Telegram Sebagai Media Informasi Seputar Handphone," *bit-Tech*, vol. 8, no. 1, pp. 33–40, Aug. 2025, doi: 10.32877/bt.v8i1.2150.

[15] E. A. Lutfi, "Analisis Usability Menggunakan Metode System Usibility Scale (SUS) Terhadap Pengguna Website SMK Muhammadiyah 6 Donomulyo, Malang," STMIK PPKIA Pradnya Paramita, Malang, Indonesia, 2024.

[16] N. Pratama, R. Anrahvi, and Stevani, "Penerapan Metode System Usability Scale (SUS) dalam Mengukur Kepuasan Mahasiswa terhadap Website Direktori Akademik," *Indonesian Journal of Business Economics and Management*, vol. 3, pp. 74–80, Jun. 2024, doi: 10.57152/ijbem.v3i2.2024.

[17] Aliyah Aliyah, Nahrun Hartono, and Asrul Azhari Muin, "Penggunaan User Acceptance Testing (UAT) Pada Pengujian Sistem Informasi Pengelolaan Keuangan Dan Inventaris Barang," *Switch : Jurnal Sains dan Teknologi Informasi*, vol. 3, no. 1, pp. 84–100, Dec. 2024, doi: 10.62951/switch.v3i1.330.

[18] C. Ulima Cahyanto and S. Wahyu, "Perancangan Aplikasi Mobile Monitoring Antar Jemput Siswa Dengan Geofencing," *JATI (Jurnal Mahasiswa Teknik Informatika)*, vol. 10, no. 1, pp. 948–956, Jan. 2026, doi: 10.36040/jati.v10i1.16918.