

IMPLEMENTASI BOT WHATSAPP UNTUK LAYANAN INFORMASI FRONTLINE (Studi Kasus: STMIK WICIDA)

Muhammad Sadam Saktia Putra[✉], Azahari, Heny Pratiwi

Teknik Informatika, STMIK Widya Cipta Dharma, Samarinda, Indonesia

Email: msadamsputra@gmail.com

ABSTRACT

This study implemented a WhatsApp bot as a frontline information service at STMIK Widya Cipta Dharma (WICIDA). The main problems were the high burden of repetitive questions, limited service hours, and inconsistent responses. WhatsApp was chosen because of its high adoption rate and support for real-time communication. The study included needs analysis, bot architecture design, Node.js-based development, knowledge base integration, and performance evaluation. The results showed that the bot was able to answer 87.4% of questions correctly, reduce staff workload by 56%, and speed up response time to <3 seconds. These findings demonstrate that the WhatsApp bot is effective as a scalable solution to improve the quality of educational information services.

Keywords: *WhatsApp Bot, Information Service, Automation.*

ABSTRAK

Penelitian ini mengimplementasikan bot WhatsApp sebagai layanan informasi frontline di STMIK Widya Cipta Dharma (WICIDA). Permasalahan utama yaitu tingginya beban pertanyaan berulang, keterbatasan jam layanan, dan ketidakkonsistenan respons. WhatsApp dipilih karena tingkat adopsinya tinggi dan mendukung komunikasi real-time. Penelitian meliputi analisis kebutuhan, perancangan arsitektur bot, pengembangan berbasis Node.js, integrasi knowledge base, dan evaluasi kinerja. Hasil menunjukkan bot mampu menjawab 87,4% pertanyaan dengan tepat, menurunkan beban staf hingga 56%, dan mempercepat waktu tanggap menjadi <3 detik. Temuan ini menunjukkan bahwa bot WhatsApp efektif sebagai solusi skalabel untuk meningkatkan kualitas layanan informasi pendidikan.

Kata Kunci: *Bot WhatsApp, Layanan Informasi, Otomatisasi.*

PENDAHULUAN

Layanan informasi merupakan salah satu komponen kunci dalam mendukung kelancaran proses akademik di perguruan tinggi. Di STMIK Widya Cipta Dharma (WICIDA), pertanyaan yang masuk ke bagian administrasi akademik (BAAK) umumnya berkisar pada alur prosedur KRS, Surat Aktif Kuliah, Cetak Transkrip Nilai dan layanan administratif lainnya. Mayoritas pertanyaan tersebut bersifat berulang terutama pada periode penerimaan mahasiswa baru di awal semester.

Berbagai penelitian di perguruan tinggi Indonesia menunjukkan bahwa *chatbot* mampu meningkatkan efisiensi layanan akademik, mempercepat waktu tanggap, dan mengurangi beban staf (Lubis & Sumartono, 2023). Implementasi *chatbot* untuk layanan informasi akademik terbukti mempermudah akses informasi bagi calon mahasiswa maupun mahasiswa aktif (Herfian & Adriansyah, 2021; Rahmawati & Susetyo, 2023).

WhatsApp dipilih sebagai platform utama karena merupakan aplikasi pesan paling populer di Indonesia dan hampir selalu aktif di ponsel mahasiswa. Studi terkait bot WhatsApp menunjukkan bahwa

platform ini efektif untuk layanan administrasi akademik, keuangan, maupun perpustakaan (Izzulhaq & Putra, 2025; Septiana & Rohman, 2025). Pada saat yang sama, penelitian lain menunjukkan bahwa kualitas interaksi dan kecepatan respons menjadi faktor penting dalam kepuasan pengguna layanan berbasis WhatsApp bot (Suryana, 2022; Widyaningrum, 2022).

TINJAUAN PUSTAKA

Chatbot adalah sistem perangkat lunak yang dirancang untuk berinteraksi dengan pengguna melalui antarmuka percakapan. Di lingkungan perguruan tinggi, chatbot banyak dimanfaatkan untuk menjawab pertanyaan seputar layanan akademik, tugas akhir, maupun pendaftaran mahasiswa baru (Arumsari, 2024; Herfian & Adriansyah, 2021; Lubis & Sumartono, 2023).

Penelitian lain memfokuskan diri pada aspek kemudahan akses dan pengalaman pengguna. Munawirah. (2025) mengembangkan chatbot berbasis Dialogflow dan NLP untuk pelayanan informasi akademik dan melaporkan akurasi jawaban yang tinggi serta tingkat kepuasan pengguna di atas 80%. Secara umum, temuan-temuan ini menguatkan bahwa *chatbot*

dapat berfungsi sebagai frontline informasi yang responsif dan konsisten.

Secara khusus, Erawan. (2024) merancang chatbot auto-reply pada WhatsApp untuk pusat informasi kampus STMIK WICIDA yang berfokus pada penyediaan jawaban instan atas pertanyaan mahasiswa terkait informasi akademik. Karya tersebut kemudian dikembangkan lebih jauh dalam bentuk tugas akhir (Erawan, 2025) dengan cakupan teknis dan evaluasi yang lebih rinci.

Dalam penelitian ini, *knowledge base* disusun dalam bentuk pasangan pertanyaan-jawaban yang dikategorikan per topik (KRS, Surat Aktif Kuliah, Cetak Transkrip Nilai). Pencocokan dilakukan dengan kombinasi kata kunci dan *similarity* sederhana sehingga mampu menoleransi variasi penulisan tertentu.

METODE PENELITIAN

Penelitian ini menggunakan pendekatan rekayasa perangkat lunak dengan tahapan umum seperti analisis kebutuhan, perancangan sistem, implementasi dan pengujian sistem. Pola tahapan ini sejalan dengan model pengembangan yang digunakan pada penelitian *chatbot* akademik lainnya (Lubis & Sumartono, 2023; Septiana & Rohman, 2025).

Analisis Kebutuhan

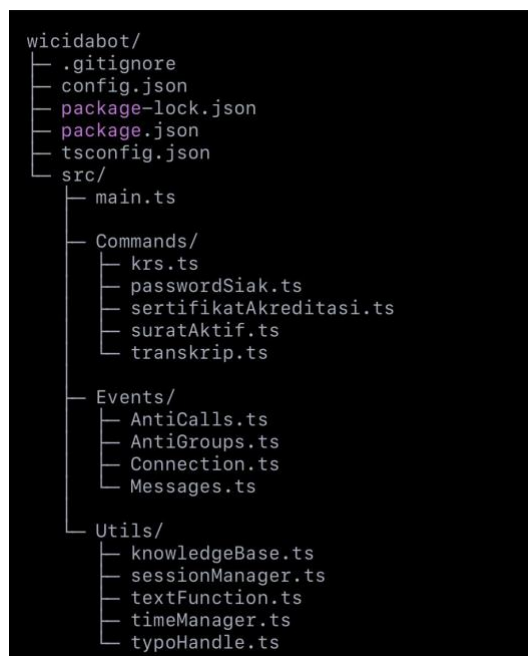
Tahap ini adalah tahap pertama yang bertujuan untuk mengidentifikasi kebutuhan layanan informasi di Bagian Administrasi Akademik dan Kemahasiswaan (BAAK) STMIK Widya Cipta Dharma. Data kebutuhan diperoleh melalui proses wawancara dengan staf BAAK yang memberikan hasil analisis bahwa pertanyaan yang masuk bersifat berulang dan dapat dikategorikan menjadi beberapa kelompok utama seperti prosedur cetak KRS, Surat Aktif Kuliah, dan Transkrip nilai.

Kebutuhan fungsional yang dirumuskan bot mampu menjawab pertanyaan umum secara otomatis berdasarkan kata kunci, fitur *anti-call* dan *auto-reject* undangan grup agar koneksi bot stabil. Sistem integrasi dengan data internal minimal berupa alamat situs dan informasi statis SIAK dan BAAK, nilai *timeout* untuk mengatur kapan waktu sesi akan berakhir dan nilai *typoThreshold* yang bertujuan untuk mengatur tingkat sensitivitas pada kesalahan penulisan.

Perancangan Sistem

Sistem dirancang menggunakan Node.js dan pustaka *Baileys WhatsApp Web API*. Arsitektur dibangun dengan menggunakan struktur modular yang bertujuan untuk lebih konsisten, seluruh modul

mengacu pada konfigurasi yang sama untuk pengembangan sistem dalam jangka panjang dan mempermudah dalam pemeliharaan sistem.



Gambar 1. Struktur Sitem

Pada tahap ini, perancangan menggunakan modul-modul pendukung seperti pemrosesan pesan untuk *parsing teks* dan normalisasi input, modul *sessionManager* yang dapat mengakhiri percakapan secara otomatis ketika tidak ada aktivitas, dan modul *typoHandle* berbasis algoritma *Levenshtein Distance* untuk menangani kesalahan penulisan pesan.

Implementasi

Tahap implementasi dimulai dari inisialisasi untuk membangun koneksi antara server Node.js dan aplikasi WhatsApp melalui pustaka *Baileys WhatsApp Web API*. Inisialisasi ini mencakup proses autentikasi serta proses penyimpanan kredensial login yang bertujuan untuk sistem dapat berjalan secara otomatis tanpa memerlukan pemindaian ulang kode QR setiap kali dijalankan.

Secara keseluruhan, inisialisasi ini menjadi fondasi utama untuk memastikan sistem mampu terhubung dengan WhatsApp secara stabil, dan mempertahankan sesi login secara berkelanjutan. Langkah selanjutnya adalah membangun sistem untuk memproses sistem respon otomatis. Bagian ini berfungsi sebagai inti dari sistem interaksi, seluruh pesan masuk dari pengguna akan diproses, dianalisis, dan diberikan tanggapan otomatis yang sesuai.

Gambar 2. Inisialisasi Sistem

```

1 import * as thisText from './Utils/textFunction'
2 import * as Config from './../config.json'
3 import { keywordIndex, loadKnowledgeBase } from './Utils/knowledgeBase'
4 import { resetTimeout, userSessions } from './Utils/sessionManager'
5 import { similarity } from './Utils/typedLands'
6
7 loadKnowledgeBase()
8
9 module.exports = {
10   name: 'messages.upsert',
11   async execute(WhatsAppClient: any, connectToWhatsApp: any, res: any) {
12     const message = res.messages[0]
13     const isMsg = message.message
14
15     if (isMsg & message.key.fromMe) return
16
17     const sender = message.key.remoteJid
18     if (!sender) return
19
20     const textMessage = (isMsg.conversation || isMsg.extendedTextMessage?.text)?.trim()
21     if (!textMessage) return
22
23     if (UserSessions[sender]) {
24       userSessions[sender] = { active: true }
25       try {
26         thisText.welcomeMessage(WhatsAppClient, sender)
27       } catch (err) {
28         console.error('Gagal kirim pesan pembuka', err)
29       }
30       resetTimeout(sender, WhatsAppClient)
31       return
32     }
33
34     resetTimeout(sender, WhatsAppClient)

```

Apabila ditemukan kecocokan, sistem akan menjalankan fungsi *execute()* dari modul terkait untuk mengirimkan jawaban kepada pengguna. Namun jika tidak ditemukan, sistem akan memanggil fungsi *notFoundKeyword()* untuk memberikan balasan standar agar percakapan tetap informatif. Secara keseluruhan, bagian pemrosesan pesan ini memungkinkan sistem beroperasi secara cerdas dan adaptif, menjaga alur komunikasi tetap alami dan

```

36 let hasAnswer = null
37 let bestScore = 0
38 const isMessage = textMessage.toLowerCase().normalize('NFKC').replace(/[^\w\s]/g, '').replace(/'/g, '').trim()
39
40 for (const [key, kb] of keywordIndex.entries()) {
41     const regex = new RegExp(`^b${key}\\b`)
42
43     if (regex.test(isMessage)) {
44         hasAnswer = kb
45         break
46     }
47
48     const score = similarity(key, isMessage)
49
50     if (score > bestScore && score >= Config.typeThreshold) {
51         hasAnswer = kb
52         bestScore = score
53     }
54 }
55
56 if (hasAnswer) return this.TextNotFoundKeyword(WhatsAppClient, sender)
57
58 try {
59     await hasAnswer.execute(WhatsAppClient, sender)
60     this.TextMoreQuestion(WhatsAppClient, sender)
61 } catch (err) {
62     console.log(`[KnowledgeBase] Error executing keyword: ${err}`)
63     this.TextNotFoundKeyword(WhatsAppClient, sender)
64 }
65 }
66 }
67 }

```

Selanjutnya membuat proses pemuatan basis pengetahuan yang berfungsi untuk memuat seluruh kata kunci secara otomatis. Sistem membaca setiap file yang berisi kata kunci, kemudian menyimpannya kedalam dua struktur data yaitu *knowledgeMap* dan *keywordIndex*. Struktur ini memungkinkan proses pencarian respons berjalan lebih cepat dan efisien dikarenakan setiap kata kunci telah dipetakan ke modul yang sesuai.

```

1 import * as fs from 'fs'
2 import path from 'path'
3
4 interface KnowledgeEntry {
5   keywords: string[]
6   execute: Function
7 }
8
9 const knowledgeMap = new Map<string, KnowledgeEntry>()
10 const keywordIndex = new Map<string, KnowledgeEntry>()
11 const knowledgeBasePath = path.join(__dirname, '..', 'Commands')
12
13 export function loadKnowledgeBase() {
14   const files = fs.readdirSync(knowledgeBasePath)
15   for (const file of files) {
16     try {
17       const modulePath = path.join(knowledgeBasePath, file)
18       const kb: KnowledgeEntry = require(modulePath)
19
20       if (!kb.keywords || !kb.execute) continue
21
22       knowledgeMap.set(file, kb)
23
24       for (const key of kb.keywords) {
25         const getKey = key.toLowerCase().normalize('NFKC').trim()
26         keywordIndex.set(getKey, kb)
27       }
28
29       console.log(`[KnowledgeBase] Loaded: ${file} (${kb.keywords.length} keywords)`)
30     } catch (err) {
31       console.log(`[KnowledgeBase] Error loading ${file}:`, err)
32     }
33   }
34 }
35
36 export { keywordIndex, knowledgeMap }

```

Kemudian Pengelolaan sesi yang berfungsi untuk mengatur waktu interaksi antara pengguna dan sistem. Modul ini menggunakan sistem penghitung waktu untuk mendeteksi aktivitas pengguna apabila pengguna tidak ada respon dalam waktu tertentu, sistem akan mengirimkan pesan penutup yang akan

menjadi sebagai tanda bahwa sistem akan mengakhiri sesi, selanjutnya akan dilakukan penghapusan data sesi agar sumber daya server tetap efisien.

Langkah selanjutnya membuat sistem koreksi kesalahan penulisan pada pengguna yang bertujuan untuk menghindari adanya kesalahan dalam penulisan pesan dari pengguna. Dengan menerapkan algoritma *Levenshtein Distance* untuk menghitung tingkat kemiripan antar kata, sehingga sistem tetap dapat memahami maksud pesan meskipun terdapat kesalahan ejaan.

```
1 import * as Config from './../config.json'
2 import * as thisText from './textFunction'
3
4 interface Session {
5   active: boolean
6   timeout?: NodeJS.Timeout
7 }
8
9 export const userSessions: Record<string, Session> = {}
10
11 // Mengatur ulang timer tiap kali user kirim pesan baru
12 export function resetTimeout(sender: string, WhatsAppClient: any) {
13   if (userSessions[sender]?.timeout) {
14     clearTimeout(userSessions[sender].timeout)
15   }
16
17   userSessions[sender].timeout = setTimeout(async () => {
18     try {
19       thisText.closingMessage(WhatsAppClient, sender)
20     } catch (err) {
21       console.error('Gagal mengirim pesan timeout:', err)
22     }
23
24     userSessions[sender].active = false
25     delete userSessions[sender]
26   }, Config.timeout * 60 * 1000)
27 }
```

Gambar 6. Pengelolaan Sesi

Secara keseluruhan, modul utilitas berperan sebagai pendukung untuk memberikan respon yang cepat, sesuai dan meningkatkan pengalaman pengguna dalam berinteraksi dengan *chatbot*.

```
1 export function handleTypo(a: string, b: string): number {
2   const m = a.length, n = b.length
3   const dp = Array.from({ length: m + 1 }, () => Array(n + 1).fill(0))
4
5   for (let i = 0; i <= m; i++) dp[i][0] = i
6   for (let j = 0; j <= n; j++) dp[0][j] = j
7
8   for (let i = 1; i <= m; i++) {
9     for (let j = 1; j <= n; j++) {
10      const cost = a[i - 1] === b[j - 1] ? 0 : 1
11
12      dp[i][j] = Math.min(
13        dp[i - 1][j] + 1,
14        dp[i][j - 1] + 1,
15        dp[i - 1][j - 1] + cost,
16      )
17    }
18  }
19
20  return dp[m][n]
21 }
22
23 export function similarity(a: string, b: string): number {
24   a = a.toLowerCase().normalize('NFKC').trim()
25   b = b.toLowerCase().normalize('NFKC').trim()
26
27   const distance = handleTypo(a, b)
28   const maxLength = Math.max(a.length, b.length)
29
30   return maxLength === 0 ? 1 : 1 - distance / maxLength
31 }
```

Gambar 7. Aloritma *Levenshtein Distance*

Bagian kata kunci berfungsi sebagai frasa yang dapat dikenali oleh sistem untuk mengatur logika pesan balasan yang akan dikirimkan kepada pengguna. Sistem akan mengenali kata yang telah terdaftar seperti “validasi krs” atau “verifikasi krs”, kemudian menjalankan fungsi yang berisi langkah-langkah panduan validasi dengan memanfaatkan data dari file *config.json* seperti alamat situs SIAK.

```
1 import * as Config from './../config.json'
2
3 module.exports = {
4   keywords: ['validasi krs', 'verifikasi krs', 'valid krs'],
5   async execute(WhatsAppClient: any, sender: any) {
6     await WhatsAppClient.sendMessage(sender, { text: 'Untuk validasi KRS, kamu bisa ikuti prosedur dibawah ini:' })
7
8     1. Kunjungi website SIAK WICIDA (S:Config.siakWebsite).
9     2. Lakukan input KRS pada menu KRS di sidebar kiri.
10    3. Temui dosen pembimbing kamu untuk melakukan bimbingan agar KRS kamu di setujui.
11    4. Kembali kunjungi website SIAK WICIDA lalu download file KRS kamu di menu KRS.
12    5. Cetak KRS menggunakan kertas Cover A4.
13    6. Datang ke loket BAAK untuk melakukan validasi.' })
14   }
15 }
```

Gambar 8. Kata Kunci

Konfigurasi sistem berfungsi sebagai pusat pengaturan utama (*Global Configuration File*) yang digunakan seluruh komponen dalam sistem. Beberapa informasi utama yang disimpan antara lain alamat situs BAAK dan SIAK, nilai parameter batas waktu sesi percakapan, dan nilai parameter untuk batas kesalahan pengetikan.

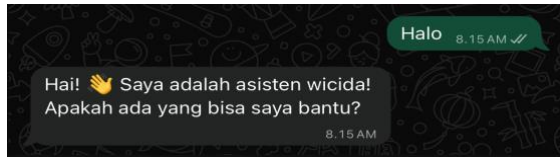
```
1 {
2   "timeout": 1,
3   "typoThreshold": 0.4,
4   "baakWebsite": "https://baak.wicida.ac.id/",
5   "siakWebsite": "https://siak.wicida.ac.id/"
6 }
```

Gambar 9. Konfigurasi Sistem

Dengan adanya konfigurasi sistem proses pembaruan informasi menjadi lebih mudah karena perubahan cukup dilakukan pada satu file tanpa perlu memodifikasi setiap modul. Sistem menjadi lebih konsisten karena seluruh modul mengaju pada konfigurasi yang sama untuk pengembangan sistem dalam jangka panjang.

Pengujian Sistem

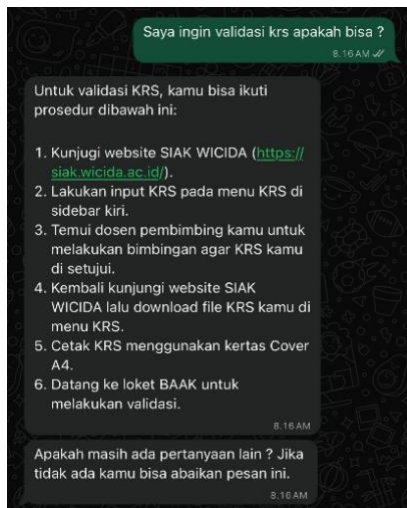
Hasil implementasi menunjukkan bahwa sistem berhasil dijalankan dengan baik menggunakan *Baileys WhatsApp Web API* sebagai jembatan komunikasi antara server dan WhatsApp. Setelah inisialisasi berhasil, sistem mampu terhubung secara stabil dan memuat seluruh modul serta kata kunci dari direktori sistem tanpa kesalahan.



Gambar 10. Implementasi Pesan Pembuka

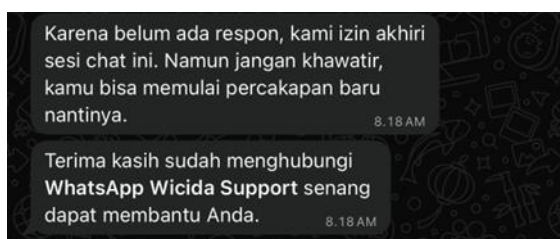
Saat pengguna mengirimkan pesan pertama, sistem akan memberikan tanggapan pembuka secara otomatis. Respon ini membuktikan bahwa fungsi pesan pembuka dapat dijalankan dengan baik. Proses ini menandakan bahwa sistem siap menerima pesan berikutnya.

Setelah pengguna menuliskan pesan yang memiliki kata kunci, sistem dapat mengenali kata kunci yang akan menghasilkan respon sesuai dengan kata kunci pengguna. Respon tersebut juga memanfaatkan nilai dari konfigurasi sistem untuk menampilkan tautan situs SIAK STMIK Widya Cipta Dharma. Setelah sistem memberikan respon yang sesuai, akan ada pesan lanjutan yang dikirimkan oleh sistem untuk mempertanyakan apakah pengguna masih memiliki pertanyaan lain atau tidak.



Gambar 11. Implementasi Kata Kunci

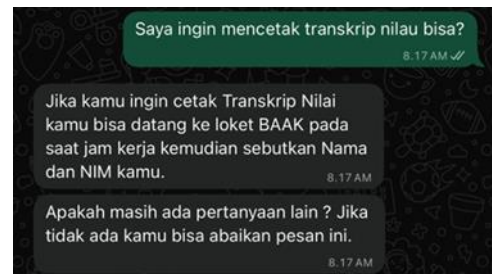
Jika pengguna tidak memiliki pertanyaan lain, pengguna dapat mengabaikan pesan dan secara otomatis sistem akan menutup sesi pembicaraan untuk pengguna dengan tujuan sistem dapat terlihat lebih natural.



Gambar 12. Implementasi Penutupan Sesi Otomatis

Ketika pengguna memberikan pesan yang memiliki kesalahan dalam pengetikan, sistem berhasil mengenali kata kunci dan memberikan respon yang sesuai. Respon ini menandakan bahwa algoritma *Levenshtein Distance* untuk menghitung tingkat kemiripan antar kata dapat berfungsi dengan baik.

Dari hasil pengujian, sistem mampu memberikan respons dalam waktu rata-rata kurang dari 3 (tiga) detik setelah pesan diterima. Hal ini menunjukkan bahwa proses pencarian kata kunci dan pemanggilan modul berjalan secara efisien. Pengujian juga membuktikan bahwa pengelolaan sesi dapat berfungsi dengan baik saat tidak ada respon dalam jangka waktu tertentu, sistem secara otomatis mengirimkan pesan penutup untuk mengakhiri percakapan.



Gambar 13. Implementasi Kesalahan Penulisan

Meskipun sistem telah berjalan dengan baik, masih terdapat peluang pengembangan lebih lanjut. Ke depannya, *chatbot* dapat dikembangkan dengan menambahkan fitur pencarian berbasis konteks menggunakan *Natural Language Processing* (NLP) agar bot mampu memahami variasi kalimat yang lebih kompleks. Selain itu, sistem dapat diintegrasikan langsung dengan basis data akademik kampus untuk memungkinkan layanan yang lebih dinamis, seperti pengecekan status pembayaran, jadwal kuliah, dan nilai mahasiswa secara *real-time*.

HASIL DAN PEMBAHASAN

Berdasarkan hasil implementasi, keunggulan bot WhatsApp sebagai layanan frontline di STMIK Widya Cipta Dharma dapat dirangkum sebagai berikut:

1. Menggunakan pendekatan *rules-based* yang stabil dan mudah dalam pemeliharaan tanpa memerlukan pelatihan model lanjutan, sistem dirancang untuk dapat dipelihara oleh admin tanpa keahlian teknis tinggi.
2. Dilengkapi fitur deteksi kesamaan (*Similarity*) untuk menangani kesalahan penulisan pesan.
3. Memiliki sistem *Anti-Call* dan *Auto-Reject* undangan grup untuk menjaga kestabilan layanan.
4. Mengurangi tekanan kerja staf dengan mengotomatisasi pertanyaan berulang.

Tabel 1. Perbandingan Kinerja Layanan

Metrik	Sebelum Bot	Sesudah Bot	Perubahan
Waktu Tanggap Rata-Rata	5-10 Menit	1,8 Detik	-98%
Akurasi Respons	-	87%	-
Jam Layanan Efektif	±8 Jam/Hari	24 Jam/Hari	±100%

Akurasi 87% menunjukkan bahwa pendekatan *rules-based* cukup memadai untuk menangani pertanyaan umum. Percepatan waktu tanggap dari orde menit menjadi orde detik mengonfirmasi bahwa adanya peningkatan kecepatan dalam layanan. Beberapa keterbatasan yang ditemukan dalam implementasi, bot masih kesulitan memahami pertanyaan yang panjang. Hal ini umum terjadi pada pendekatan *rule-based* dan menjadi alasan pengembangan ke arah NLP lebih lanjut (Arumsari, 2024; Munawirah, 2025).

Keterbatasan lainnya adalah ketergantungan pada kualitas *Knowledge Base*, jika *Knowledge Base* tidak diperbaharui ketika ada kebijakan baru. Bot akan berpotensi memberikan informasi yang tidak lagi relevan. Bot juga tidak terintegrasi penuh dengan sistem akademik, sehingga belum dapat memberikan informasi personal seperti status KRS atau nilai mahasiswa.

KESIMPULAN

Penelitian ini mengimplementasikan bot WhatsApp sebagai layanan informasi frontline di STMIK Widya Cipta Dharma. Berdasarkan hasil evaluasi, dapat disimpulkan bahwa:

1. Bot mampu meningkatkan kecepatan layanan secara signifikan, menurunkan waktu tanggap dari 5-10 menit menjadi sekitar 1,8 detik.
2. Bot mencapai akurasi respons sebesar 87%, cukup tinggi untuk konteks pertanyaan berulang terstruktur.
3. Jam layanan efektif bertambah dari sekitar 8 jam per hari menjadi 24 jam penuh, sehingga akses informasi bagi mahasiswa meningkat.

Selain itu, sistem berhasil menampilkan perilaku interaktif yang menyerupai komunikasi manusia. Bot dapat memberikan sapaan pembuka, menutup percakapan secara sopan, serta menanyakan apakah pengguna masih membutuhkan bantuan tambahan. Dengan demikian, pengalaman pengguna menjadi lebih alami dan nyaman, sekaligus meningkatkan efektivitas penyampaian informasi di lingkungan kampus.

Meskipun sistem telah berjalan dengan baik, masih terdapat peluang pengembangan lebih lanjut. Ke depannya, chatbot dapat dikembangkan dengan menambahkan fitur pencarian berbasis konteks menggunakan *Natural Language Processing* (NLP) agar bot mampu memahami variasi kalimat yang lebih kompleks. Selain itu, sistem dapat diintegrasikan langsung dengan basis data akademik kampus untuk memungkinkan layanan yang lebih dinamis, seperti jadwal kuliah dan nilai mahasiswa secara *real-time*.

Secara keseluruhan, implementasi menunjukkan bahwa penerapan chatbot akademik berbasis WhatsApp merupakan solusi efektif dalam mendukung digitalisasi layanan administrasi pendidikan. Sistem ini berpotensi menjadi fondasi bagi pengembangan layanan cerdas lainnya yang mendukung efisiensi dan transparansi informasi akademik di perguruan tinggi.

DAFTAR PUSTAKA

- Arumsari, D., Kharisma, & Aesyi, U. S. (2024). Sistem chatbot layanan informasi mahasiswa menggunakan algoritma Long Short-Term Memory. *Indonesian Journal on Data Science*, 2(2), 77–86.
- Bariah, S. H., Pratiwi, W., & Imania, K. A. N. (2022). Pengembangan virtual assistant chatbot berbasis WhatsApp pada pusat layanan informasi mahasiswa Institut Pendidikan Indonesia–Garut. *Jurnal PETIK*, 8(1), 66–79.
- Darmawan, M. F., Sujono, Santosa, A. L., Khasan, A. F., Misbakhur Surur, M., Jamaluddukha, M., & Basri, M. F. (2024). Pembuatan WhatsApp bot untuk sistem informasi pelayanan Desa Kromong Ngusikan Jombang. *Informatika: Jurnal Pengabdian Masyarakat*, 5(3), 196–199.
- Erawan, Y., Yusnita, A., & Adytia, P. (2024). Perancangan chatbot auto-reply pada WhatsApp untuk mendukung pusat informasi kampus STMIK Wicida. *Sebatik*, 28(2).
- Herfian, M. R., & Adriansyah, A. R. (2021). Analisis dan perancangan aplikasi chatbot dalam pelayanan penerimaan mahasiswa baru pada perguruan tinggi. *Jurnal Informatika Terpadu*, 7(2).
- Izzulhaq, M. R., & Putra, R. R. J. (2025). Rancang bangun sistem informasi perpustakaan terintegrasi chatbot WhatsApp. *Digital Transformation Technology*, 5(1), 66–75.
- Lubis, A., & Sumartono, I. (2023). Implementasi layanan akademik berbasis chatbot untuk meningkatkan interaksi mahasiswa. *RESOLUSI: Rekayasa Teknik Informatika dan Informasi*, 3(5), 264–270.
- Munawirah, M., Mardjani, E., & Kristian, K. (2025). Aplikasi chatbot berbasis Dialogflow dan NLP untuk pelayanan informasi akademik pada Fakultas Ilmu Komputer Universitas Tomakaka. *Jurnal Ilmiah Multidisiplin Amsir*, 3(2).

- Prasetyo, S. E., Puteri, V. A., & Sabariman. (2025). Implementasi chatbot AI dan WhatsApp untuk mendukung penjualan dan konsultasi rakitan komputer di Toko Micro Batam. *Journal of Information System Management*, 7(1), 120–126.
- Puteri, A. N. (2022). Aplikasi chatbot untuk layanan informasi penerimaan calon mahasiswa baru. *Prosiding Seminar Nasional Teknik Elektro dan Informatika (SNTEI)*.
- Rahmawati, & Susetyo. (2023). Pemanfaatan chatbot untuk informasi pendaftaran calon mahasiswa. *IT-EXPLORE*, 2(2), 89–103.
- Saputra, F., Handoko, R. M., Putra, W., Priskila, R., & Pranatawijaya, V. H. (2024). Chatbot berbasis WhatsApp Teknik Informatika Universitas Palangkaraya: Rules based system. *Jurnal Teknologi Informatika dan Komputer MH. Thamrin*, 10(1), 296–308.
- Septiana, Y., & Rohman, M. I. F. (2025). Rancang bangun sistem WhatsApp bot layanan administrasi akademik dan keuangan di Institut Teknologi Garut menggunakan metode Rational Unified Process. *Jurnal Algoritma*, 22(2), 1234–1242.
- Situmeang, R. N., Purba, E. N., & Harianja, E. J. G. (2023, November). Designing Chatbot About Tourism Recommendations In North Sumatra. In *2023 International Conference of Computer Science and Information Technology (ICOSNIKOM)* (pp. 1-4). IEEE.
- Suryana, R., Aryanto, M., Kurniawan, R., Satmata, K. S. G. P., Yulianti, Y., & Saifudin, A. (2022). Pengembangan kecerdasan buatan WhatsApp chatbot untuk mahasiswa. *Jurnal Teknologi Sistem Informasi dan Aplikasi*, 5(1), 37–46.
- Widyaningrum, E. A. (2022). Analisis pengaruh kualitas layanan informasi swamedikasi online berbasis WhatsApp bot. *Jurnal Farmasi Indonesia*, 14(2), 114–123.