Jurnal Ilmu Komputer dan Sistem Informasi (JIRSI) **e-ISSN 2830-3954**
Volume: 5, Nomor: 2, Mei 2026: 400-413 **p-ISSN 2830-6031**
https://jurnal.unity-academy.sch.id/index.php/jirsi/index

Perancangan Sistem Asisten Produktivitas Pribadi Cerdas "Prioritas Bot" Menggunakan Large Language Model (LLM) Berbasis N8N dan Telegram

Design of an Intelligent Personal Productivity Assistant System "Priority Bot" Using a Large Language Model (LLM) Based on N8N and Telegram

Elita Nur Ilahi*[1], Muhammad Encep[2], Setyono[3] (* corespondent author)
[1,2,3]Program Studi lmu Komputer, Fakultas Ilmu Komputer, Universitas Djuanda
E-mail: [1]i.2210060@unida.ac.id, [2]ahmadpoac@unida.ac.id, [3]setyono@unida.ac.id

Abstrak

Kompleksitas manajemen tugas dan alur kerja di era digital menuntut adanya alat bantu cerdas. Penelitian ini merancang Sistem Asisten Produktivitas Pribadi Cerdas yang mengintegrasikan tiga komponen utama: (1) Large Language Model (LLM) GPT-4o yang berfungsi sebagai inti pemrosesan untuk pemahaman bahasa alami dan generasi respons kontekstual; (2) Platform otomasi low-code n8n yang bertindak sebagai orkestrator, mengelola alur kerja dan menghubungkan LLM dengan API eksternal seperti Google Calendar; serta (3) Telegram Bot yang digunakan sebagai antarmuka pengguna utama karena aksesibilitasnya. Metode perancangan menggunakan pendekatan Prototyping adaptasi Pressman & Maxim (2020) untuk pengembangan iteratif dan validasi fungsional. Pengujian fungsionalitas dilakukan dengan Black Box Testing, sedangkan tingkat kegunaan diukur menggunakan System Usability Scale (SUS) yang melibatkan 20 responden mahasiswa aktif di wilayah Bogor yang dipilih melalui Purposive Sampling. Hasil penelitian berupa prototipe fungsional yang mampu mengelola jadwal secara end-to-end melalui percakapan bahasa alami. Sistem memperoleh skor SUS rata-rata 88,5 dari 20 responden yang masuk kategori Excellent dan peringkat Grade A, mengindikasikan potensi signifikan integrasi LLM dan otomasi alur kerja untuk meningkatkan produktivitas pribadi secara efektif.

***Kata kunci**: Asisten Produktivitas, Large Language Model (LLM), n8n, Otomatisasi Alur Kerja, Telegram Bot, Prototyping*

Abstract

The complexity of task and workflow management in the digital era necessitates intelligent assistive tools. This study designs an Intelligent Personal Productivity Assistant System by integrating three primary components: (1) A Large Language Model (LLM) GPT-4o serving as the processing core for natural language understanding and contextual response generation; (2) The n8n low-code automation platform, acting as an orchestrator to manage workflows and connect the LLM to external APIs such as Google Calendar; and (3) A Telegram Bot utilized as the primary user interface due to its high accessibility. The design methodology employs a Prototyping approach adapted from Pressman & Maxim (2020), enabling iterative development and functional validation. Black Box Testing was applied for functional evaluation, while the System Usability Scale (SUS) was used to measure usability, involving 20 active university students in the Bogor region selected through Purposive Sampling. The research resulted in a functional prototype capable of managing schedules end-to-end via conversational natural language. The system achieved an average SUS score of 88.5 from 20 respondents, classified as Excellent with a Grade A rating, indicating the significant potential of integrating LLMs and workflow automation to effectively enhance personal productivity.

***Keywords**: Productivity Assistant, Large Language Model (LLM), n8n, Workflow Automation, Telegram Bot, Prototyping*

1. PENDAHULUAN

Pengelolaan aktivitas dan jadwal harian merupakan aspek fundamental dalam mencapai efisiensi kerja di kehidupan modern. Ketika intensitas agenda melampaui daya ingat manusia, individu memerlukan alat bantu untuk mengorganisir tugas. Secara konvensional, pencatatan dilakukan manual menggunakan media fisik atau fitur kalender statis. Namun, metode ini kurang relevan bagi tanggung jawab yang kompleks karena keterbatasan dalam memberikan pengingat aktif dan ketiadaan integrasi dengan sumber informasi digital [1].

Di era digital, tantangan manajemen produktivitas pribadi semakin dipicu oleh fenomena *information overload*. Individu terus terpapar arus data masif dari berbagai platform, seperti email korporasi, pesan instan, hingga perangkat lunak manajemen proyek [2]. Meskipun tersedia beragam aplikasi produktivitas, penggunaan banyak platform yang tidak terhubung justru memicu beban kognitif (*cognitive load*) baru yang kontraproduktif [3]. Permasalahan utama yang muncul adalah fragmentasi alur kerja, di mana pengguna terpaksa melakukan *context switching* secara berulang untuk sinkronisasi data manual antara aplikasi seperti WhatsApp, email, dan Notion.

Kondisi ini dirasakan secara nyata di lingkungan akademik, di mana mahasiswa harus menyeimbangkan jadwal kuliah, tenggat tugas, dan aktivitas organisasi yang dinamis. Kegagalan manajemen waktu yang sistematis terbukti berkorelasi langsung dengan penurunan performa akademik [4]. Di sisi lain, muncul hambatan berupa *app fatigue*, yakni kejenuhan pengguna untuk mengunduh aplikasi baru sehingga aplikasi produktivitas yang berdiri sendiri sering kali diabaikan [5], [6].

Melihat pola perilaku saat ini, penggunaan aplikasi pesan instan sebagai platform utama komunikasi menunjukkan tingkat keterlibatan pengguna (*user engagement*) yang sangat tinggi [7]. Penelitian ini mengusulkan integrasi tiga pilar teknologi utama: Telegram sebagai Conversational UI dengan aksesibilitas tinggi [8], Kecerdasan Buatan multimodal dari OpenAI (GPT-4o) untuk penalaran dan ekstraksi data terstruktur [9], serta platform otomasi *low-code* n8n sebagai jembatan interkonektivitas API [10].

Meskipun ketiga teknologi tersebut sudah matang secara fungsional, integrasinya masih terkendala saat dioperasikan secara terpisah. Model GPT-4o memiliki kemampuan pemahaman bahasa alami yang kuat, namun tidak dapat mengeksekusi tindakan langsung pada sistem kalender eksternal secara mandiri [9]. Sebaliknya, platform otomasi n8n sangat unggul dalam eksekusi protokol API, tetapi instruksi yang diterimanya bersifat kaku dan memerlukan input yang sudah terstruktur [10].

Berangkat dari identifikasi masalah tersebut, penelitian ini bertujuan merancang Sistem Asisten Produktivitas Pribadi Cerdas “Prioritas Bot”. Dalam sistem ini, LLM diposisikan sebagai processing unit yang menerjemahkan niat (intent) pengguna menjadi tindakan digital pada layanan pihak ketiga, sementara n8n berperan sebagai orkestrator dinamis yang menjembatani transmisi data antara Telegram Bot API, OpenAI API, dan Google Calendar API. Pendekatan orkestrasi ini krusial untuk meminimalisir distorsi informasi atau kesalahan interpretasi (halusinasi) yang sering terjadi pada model bahasa besar dalam skenario mandiri [11]. .

2. METODOLOGI PENELITIAN

Penelitian ini menggunakan pendekatan Penelitian Terapan (*Applied Research*) dengan skema Rancang Bangun (*Design and Build*) yang selaras dengan paradigma *Design Science Research* (DSR) dalam bidang sistem informasi [12]. Strategi pengembangan sistem yang diterapkan adalah Metode *Prototyping* yang mengacu pada kerangka kerja Pressman & Maxim (2020) [13].

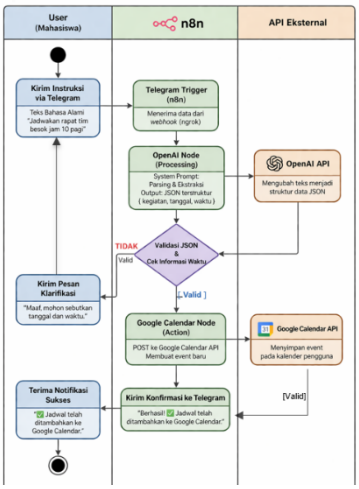
Pemilihan metode *Prototyping* dinilai strategis mengingat karakteristik sistem yang berbasis *Large Language Model* (LLM) bersifat non-deterministik. Dalam sistem seperti ini, spesifikasi teknis tidak dapat ditetapkan secara kaku pada awal pengembangan. Penerapan metode ini memungkinkan peneliti melakukan siklus iteratif intensif, di mana umpan balik pengguna digunakan untuk menyempurnakan system prompt pada OpenAI dan logika alur kerja pada n8n.

2.1. Objek dan Lokasi Penelitian

Objek penelitian ini berfokus pada perancangan sistem asisten produktivitas cerdas yang memanfaatkan integrasi LLM OpenAI GPT-4o dan platform orkestrasi n8n. Subjek penelitian adalah mahasiswa aktif yang berdomisili di wilayah Kabupaten dan Kota Bogor. Pengambilan sampel menggunakan teknik Purposive Sampling dengan kriteria inklusi mahasiswa yang memiliki intensitas kegiatan tinggi, guna memastikan urgensi terhadap alat bantu manajemen waktu [14].

2.2. Arsitektur Sistem

Arsitektur sistem Prioritas Bot dibangun dengan mengintegrasikan tiga komponen utama yang bekerja secara sinergis. Pertama, Telegram Bot API berfungsi sebagai antarmuka pengguna yang menangkap input bahasa alami dan menyampaikan konfirmasi kepada pengguna. Kedua, platform n8n berperan sebagai orkestrator alur kerja berbasis *node* yang menghubungkan seluruh layanan API. Ketiga, model GPT-4o dari OpenAI berperan sebagai unit pemroses kognitif yang melakukan *intent recognition* dan *entity extraction* [9]. Infrastruktur pendukung terdiri dari Docker Desktop untuk kontainerisasi n8n, ngrok untuk *secure tunneling webhook*, dan Laragon sebagai lingkungan pengembangan lokal [15], [16]. Berikut alur sistem ditunjukkan oleh gambar 1.



Gambar 1. Arsitektur Sistem Prioritas Bot

Alur kerja (workflow) sistem berjalan secara sekuensial: (1) pengguna mengirim instruksi bahasa alami via Telegram; (2) Telegram Trigger Node pada n8n menerima *payload* melalui *webhook ngrok*; (3) AI Agent Node memproses teks menggunakan GPT-4o untuk mengekstraksi *intent* dan entitas ke format JSON; (4) Decision Node melakukan validasi data JSON; (5) Google Calendar Node mengeksekusi operasi API; (6) konfirmasi dikirimkan kembali ke pengguna melalui Telegram.

2.3. Metode Pengujian

Pengujian kelayakan sistem dibatasi pada dua metode utama. Pertama, pengujian fungsional menggunakan *Black Box Testing* untuk memvalidasi keberhasilan alur kerja *end-to-end* dan memastikan tidak ada distorsi informasi dari LLM saat mengeksekusi API. Evaluasi ini merujuk pada prinsip rekayasa perangkat lunak oleh Pressman & Maxim (2020) [13]. Kedua, pengujian kegunaan (*usability*) menggunakan instrumen *System Usability Scale* (SUS) yang dikembangkan oleh Brooke (1996) [17], untuk mengukur tingkat kemudahan, penerimaan pengguna, dan efisiensi sistem.

Evaluasi usability melibatkan 20 responden mahasiswa aktif di wilayah Bogor yang dipilih melalui *purposive sampling*. Penentuan jumlah 20 responden didasarkan pada landasan teoritis Virzi (1992) dan Nielsen (2000) yang menyatakan bahwa 5 hingga 20 responden sudah cukup untuk mendeteksi lebih dari 90% masalah kegunaan utama [18], [19]. Perhitungan skor SUS mengikuti rumus standar John Brooke (1996) dengan hasil dalam rentang 0–100.

3. HASIL DAN PEMBAHASAN

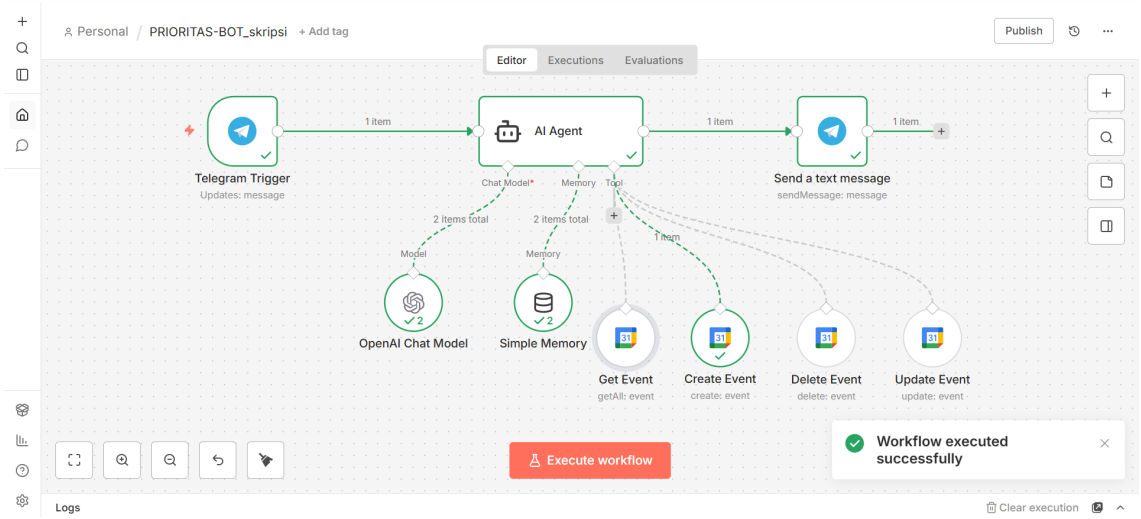
Bagian ini memaparkan hasil implementasi dan pengujian sistem asisten produktivitas cerdas “Prioritas Bot” secara menyeluruh. Pembahasan diorganisasikan ke dalam lima sub-bagian yang saling berkesinambungan. Sub-bagian 3.1 menguraikan realisasi infrastruktur teknis dan antarmuka pengguna sebagai fondasi operasional sistem. Sub-bagian 3.2 menganalisis proses pemrosesan bahasa alami dan mekanisme ekstraksi entitas yang dilakukan oleh model GPT-4o. Sub-bagian 3.3 menyajikan hasil pengujian fungsionalitas sistem secara menyeluruh melalui metode Black Box Testing, mencakup skenario operasi normal maupun penanganan kesalahan. Sub-bagian 3.4 membahas hasil pengujian kemampuan pemahaman bahasa alami (Natural Language Processing) terhadap berbagai variasi gaya bahasa pengguna. Terakhir, sub-bagian 3.5 menyajikan hasil evaluasi tingkat kegunaan (usability) sistem menggunakan instrumen System Usability Scale (SUS) beserta analisis persepsi pengguna secara kuantitatif.

3.1. Infrastruktur dan Antarmuka

Sebelum mengimplementasikan alur kerja, peneliti membangun fondasi infrastruktur teknis yang stabil. Instance n8n dijalankan di dalam kontainer Docker yang terkonfigurasi dengan volume penyimpanan data yang persisten. Layanan ngrok digunakan untuk mengekspos endpoint webhook ke internet publik melalui jalur terenkripsi (HTTPS), memungkinkan Telegram Bot API mengirimkan payload secara real-time ke server lokal. Antarmuka pengguna direalisasikan melalui

e-ISSN 2830-3954
p-ISSN 2830-6031

Telegram Bot yang memiliki identitas visual berupa logo dan nama resmi "Prioritas Bot", sehingga meningkatkan kepercayaan pengguna.



Gambar 2. Alur Kerja/*Workflow* N8N

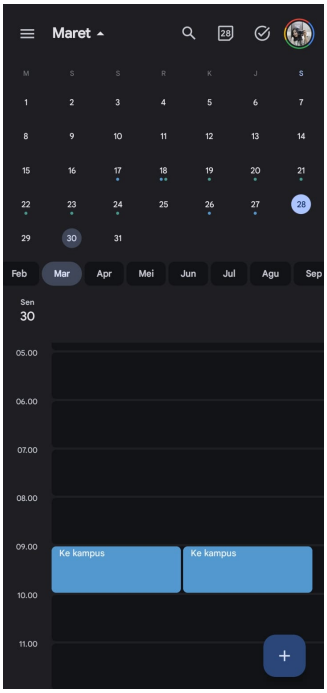
Gambar 2 menunjukkan visualisasi alur kerja (*workflow*) pada platform n8n yang telah berhasil dikonfigurasi. Sistem bekerja dengan arsitektur berbasis kejadian (*event-driven*), di mana proses baru berjalan hanya ketika pengguna mengirimkan pesan melalui Telegram. Status "*Workflow executed successfully*" yang terlihat pada gambar membuktikan bahwa seluruh rangkaian integrasi dari pesan masuk hingga jadwal tersimpan di Google Calendar berjalan sinkron tanpa error, memvalidasi bahwa platform *low-code* seperti n8n mampu mempermudah penggabungan berbagai layanan API secara efektif [10].

3.2. Analisis Pemrosesan Bahasa Alami dan Ekstraksi Entitas

Analisis Pemrosesan Bahasa Alami dan Ekstraksi Entitas

Tahap krusial dalam mekanisme Prioritas Bot terletak pada proses penalaran dan ekstraksi data oleh model GPT-4o. Berdasarkan analisis pada *node* Simple Memory, sistem mendokumentasikan pemanggilan fungsi melalui perintah tool_calls. Atribut Summary terisi dengan nilai "Ke kampus", membuktikan keberhasilan AI mengidentifikasi subjek kegiatan utama dari pesan teks yang dikirim pengguna.

Sistem menunjukkan presisi tinggi dalam menerjemahkan waktu relatif. Instruksi pengguna yang menyebutkan "Senin jam sembilan pagi" secara otomatis dikonversi oleh GPT-4o menjadi format stempel waktu standar ISO 8601 (2026-03-30T09:00:00+07:00). Penggunaan zona waktu +07:00 membuktikan bahwa sistem telah dikonfigurasi secara tepat untuk wilayah waktu Indonesia Barat (WIB) [9].



Gambar 3. Sinkronisasi pada Google Calendar

Gambar 3 menunjukkan hasil nyata dari proses integrasi, di mana entri jadwal berjudul "Ke kampus" muncul secara otomatis pada tanggal 30 Maret 2026 pukul 09.00 WIB. Hal ini membuktikan bahwa orkestrator n8n mampu mendistribusikan *payload* dari unit logika AI menuju endpoint API tujuan tanpa kerusakan data atau kehilangan konteks informasi [11].

3.3. Hasil Pengujian Black Box Testing

Pengujian *Black Box Testing* dilakukan untuk memvalidasi bahwa setiap fungsi spesifik pada Prioritas Bot berjalan sesuai skenario yang diharapkan. Tabel 1 merangkum hasil pengujian fungsionalitas utama sistem.

Tabel 1. Hasil Pengujian *Black Box* Sistem Prioritas Bot

No	Fungsi yang Diuji	Hasil yang Diharapkan	Status
1	Aktivasi *Webhook*	n8n menerima *payload* dan memicu *workflow* secara *real-time*	Berhasil
2	Interpretasi Niat (*Intent*)	OpenAI GPT-4o mengenali niat sebagai pembuatan jadwal (*Create Event*)	Berhasil
3	Ekstraksi Entitas	Sistem mengonversi waktu menjadi format ISO 8601 yang presisi	Berhasil
4	Manajemen Memori	Bot mempertahankan konteks percakapan sebelumnya melalui Simple Memory	Berhasil

No	Fungsi yang Diuji	Hasil yang Diharapkan	Status
5	Integrasi API Google Calendar	Jadwal muncul di kalender sesuai subjek dan waktu yang ditentukan	Berhasil
6	Konfirmasi Output	Pengguna menerima notifikasi teks konfirmasi keberhasilan di Telegram	Berhasil
7	Keamanan OAuth 2.0	Koneksi API terjaga tanpa autentikasi ulang secara manual	Berhasil

Berdasarkan data pada Tabel 1, seluruh butir pengujian menunjukkan status Berhasil. Hal ini mengonfirmasi bahwa integrasi antara Telegram Bot API, orkestrator n8n, dan model bahasa GPT-4o telah mencapai tingkat sinkronisasi yang optimal. Kemampuan sistem dalam menangani rujukan waktu relatif dan mempertahankan konteks memori membuktikan bahwa Prioritas Bot mampu meminimalisir hambatan teknis yang biasanya ditemukan pada sistem penjadwalan konvensional yang bersifat kaku [10], [20].

Pengujian penanganan kesalahan (*error handling*) juga dilakukan untuk memvalidasi ketahanan sistem. Prioritas Bot telah mengimplementasikan prinsip *Graceful Degradation*, di mana sistem tidak langsung berhenti saat menghadapi kesalahan, melainkan memberikan respons yang informatif. Saat menghadapi instruksi ambigu, model GPT-4o dikonfigurasi melalui system prompt untuk tidak menebak-nebak (*hallucinate*) data yang kosong, melainkan secara aktif memandu pengguna melengkapi parameter yang kurang.

Berdasarkan data pada Tabel 1, seluruh butir pengujian menunjukkan status Berhasil. Hal ini mengonfirmasi bahwa integrasi antara Telegram Bot API, orkestrator n8n, dan model bahasa GPT-4o telah mencapai tingkat sinkronisasi yang optimal. Kemampuan sistem dalam menangani rujukan waktu relatif dan mempertahankan konteks memori membuktikan bahwa Prioritas Bot mampu meminimalisir hambatan teknis yang biasanya ditemukan pada sistem penjadwalan konvensional yang bersifat kaku [10], [20].

Pengujian ini juga memvalidasi bahwa infrastruktur pendukung seperti ngrok sebagai jalur *tunneling* dan Docker sebagai lingkungan kontainer telah dikonfigurasi dengan tepat, sehingga latensi pengiriman data tetap berada dalam ambang batas yang dapat diterima oleh pengguna [15], [23]. Dengan tuntasnya tahap pengujian fungsional ini, sistem dinyatakan layak untuk masuk ke tahap pengujian subjektif melalui persepsi pengguna (*Usability Testing*).

1. Pengujian Fungsionalitas Operasi Kalender (CRUD *Testing*)

Pengujian ini bertujuan untuk memastikan bahwa Prioritas Bot mampu menangani berbagai instruksi perubahan jadwal yang dinamis dari pengguna. Hasil pengujian ini dipaparkan pada Tabel 2 berikut:

Tabel 2. Pengujian *Black Box* CRUD Sistem Prioritas Bot

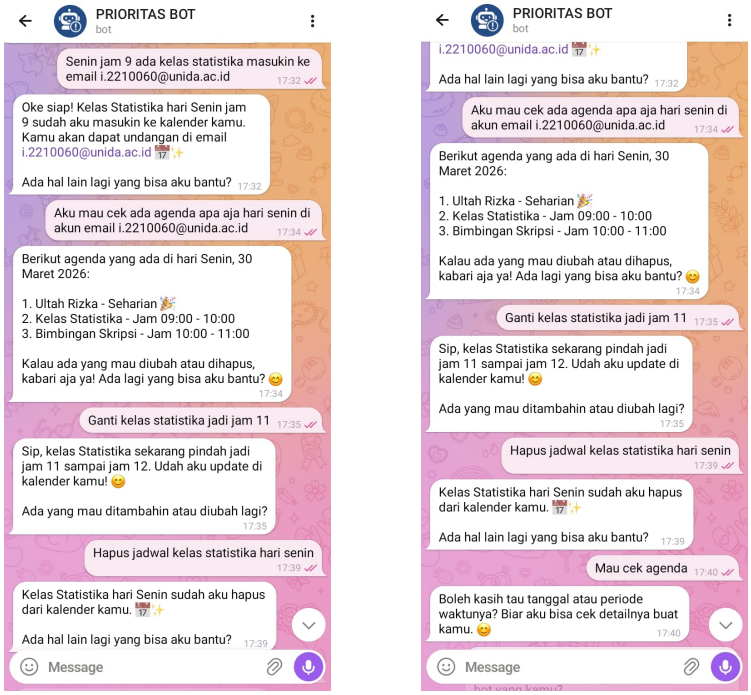
No	Operasi API	Skenario Pengujian (Input Telegram)	Hasil yang Diharapkan	Status
1	*Create Event*	"Senin jam 9 ada kelas Statistika"	Jadwal baru bernama "Statistika" muncul di kalender sesuai waktu.	Berhasil
2	*Get Event*	"Ada agenda apa hari Senin?"	Bot merespons dengan rincian jadwal yang ada pada tanggal tersebut.	Berhasil
3	*Update Event*	"Ganti kelas Statistika jadi jam 11"	Waktu pada agenda "Statistika" di kalender bergeser otomatis ke jam 11.	Berhasil
4	*Update Event*	"Ubah nama agenda Statistika jadi Ujian Tengah Semester"	Judul agenda di kalender berubah tanpa mengubah slot waktunya.	Berhasil
5	*Delete Event*	"Hapus jadwal kelas Statistika hari Senin"	Agenda tersebut terhapus secara permanen dari Google Calendar.	Berhasil
6	*Error Handling*	Mengirim pesan yang tidak mengandung informasi waktu atau subjek.	Bot memberikan pesan klarifikasi atau meminta informasi tambahan.	Berhasil

Berdasarkan hasil pengujian pada Tabel 2, sistem telah mampu menjalankan fungsi pengelolaan agenda secara komprehensif. Keberhasilan pada fungsi *Update* dan *Delete* menunjukkan bahwa AI Agent pada n8n memiliki kemampuan untuk mencari ID event yang spesifik di Google Calendar terlebih dahulu sebelum melakukan modifikasi data. Hal ini membuktikan efektivitas penggunaan Tools pada arsitektur agen cerdas yang dikembangkan [23].
Kemampuan bot dalam melakukan fungsi Get Event juga sangat membantu mahasiswa dalam melakukan konfirmasi jadwal tanpa harus membuka aplikasi kalender secara manual. Hal ini sejalan dengan tujuan penelitian untuk mereduksi app fatigue dengan memusatkan seluruh kegiatan manajemen waktu di dalam satu antarmuka pesan instan saja [6], [11].

Pengujian fungsionalitas sistem lebih lanjut dilakukan dengan menerapkan skenario percakapan beruntun (*multi-turn conversation*). Skenario ini dirancang

e-ISSN 2830-3954
p-ISSN 2830-6031

untuk menguji bagaimana sistem merespons berbagai jenis instruksi manajemen kalender dalam satu alur percakapan yang berkelanjutan. Instruksi yang diujikan mencakup proses penambahan jadwal baru, pembacaan daftar agenda, pengubahan waktu jadwal, penghapusan jadwal, serta pengujian respons sistem terhadap masukan (*input*) yang tidak lengkap. Bukti interaksi antara pengguna dan asisten produktivitas melalui antarmuka Telegram pada skenario tersebut disajikan pada Gambar 4 berikut:

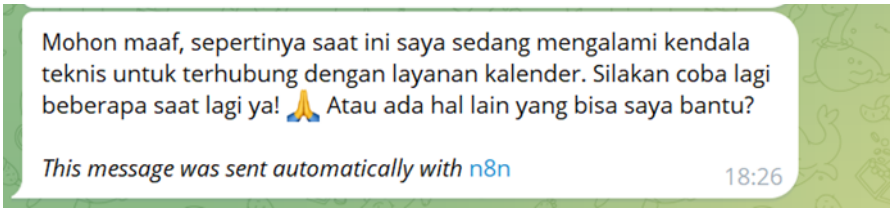


Gambar 4. Skenario Pengujian Interaksi Manajemen Jadwal dan Penanganan Ambiguitas

Berdasarkan Gambar 4, percakapan dimulai dengan instruksi pengguna untuk menjadwalkan "kelas statistika" pada hari Senin pukul 09:00 ke alamat email spesifik, yang langsung direspons dengan konfirmasi penyisipan jadwal oleh sistem. Selanjutnya, pengguna meminta sistem untuk memeriksa agenda pada hari Senin di email yang sama. Sistem merespons dengan menampilkan daftar tiga kegiatan beserta rentang waktunya (Ultah Rizka, Kelas Statistika, dan Bimbingan Skripsi).
Pada tahap pengujian modifikasi data, pengguna menginstruksikan sistem untuk mengganti waktu kelas statistika menjadi pukul 11. Sistem memproses instruksi tersebut dan membalas bahwa jadwal telah diperbarui menjadi pukul 11:00 sampai 12:00. Alur dilanjutkan dengan instruksi penghapusan jadwal, di mana perintah "Hapus jadwal kelas statistika hari senin" berhasil dieksekusi dan dikonfirmasi oleh sistem.
Pada bagian akhir percakapan (sisi kanan gambar), dilakukan pengujian penanganan kesalahan (*error handling*) dengan memberikan instruksi yang bersifat ambigu, yaitu "Mau cek agenda". Karena instruksi tersebut tidak memuat parameter waktu yang spesifik, sistem tidak melakukan eksekusi pencarian secara acak. Sebaliknya, sistem memicu mekanisme fallback dengan memberikan pertanyaan

balasan: "Boleh kasih tau tanggal atau periode waktunya? Biar aku bisa cek detailnya buat kamu." Respons ini menunjukkan bahwa sistem mampu mengidentifikasi kekurangan parameter pada instruksi bahasa alami dan secara aktif meminta klarifikasi kepada pengguna sebelum melakukan pemrosesan data ke kalender.

Selain pengujian fungsionalitas pada kondisi normal, evaluasi juga dilakukan untuk menguji keandalan sistem (*reliability*) saat menghadapi kendala teknis. Skenario ini menguji bagaimana sistem merespons apabila terjadi gangguan seperti terputusnya koneksi API layanan pihak ketiga (API Error) atau kegagalan pemrosesan data pada node kalender. Tujuan dari pengujian ini adalah untuk memastikan bahwa mekanisme penanganan kesalahan (*error handling*) berfungsi dengan baik dan sistem tidak berhenti beroperasi secara tiba-tiba tanpa memberikan informasi kepada pengguna. Hasil respons sistem saat simulasi kendala teknis terjadi disajikan pada Gambar 5 berikut:



Gambar 5. Pesan Balasan Sistem (*Fallback*) saat Terjadi Kendala Teknis

Berdasarkan Gambar 5, sistem terbukti mampu mendeteksi adanya kegagalan proses dan langsung meresponsnya dengan mengirimkan pesan darurat (*fallback message*).

3.4. Hasil Pengujian Hasil Pengujian Bahasa Alami (NLP)

Pengujian pemahaman bahasa alami dilakukan melalui observasi kualitatif terhadap kemampuan *Prompt Engineering* dalam menangani variasi gaya bahasa pengguna. Tabel 3 merangkum hasil pengujian.

Tabel 3. Hasil Pengujian Bahasa Alami (NLP)

No	Kategori	Contoh Input	Status
1	Bahasa Formal	"Jadwalkan rapat evaluasi hari Senin pukul 09.00 WIB."	Berhasil
2	Bahasa Gaul/Singkatan	"bsk ngampus jm 9"	Berhasil
3	Kalimat *Typo*/Salah Ketik	"ingetiin bsook jm dua ada kulyah"	Berhasil

Kombinasi antara model GPT-4o dan manajemen memori pada orkestrator n8n berhasil menciptakan asisten digital yang tidak kaku, sangat adaptif terhadap gaya

komunikasi mahasiswa, dan mampu mempertahankan konteks percakapan secara berkelanjutan. Berikut pengujian visual terhadap *prompt engineering*:

Gambar 6. Pengujian Bahasa Alami Pengguna

Karena model yang digunakan adalah *pre-trained* LLM, pengujian performa pemrosesan bahasa tidak menggunakan metrik kuantitatif (*Machine Learning metrics*), melainkan melalui observasi kualitatif terhadap kemampuan *Prompt Engineering* dalam menangani variasi gaya bahasa pengguna. Pengujian dilakukan dengan mensimulasikan percakapan nyata yang mencakup bahasa formal, singkatan (bahasa sehari-hari), dan kesalahan pengetikan (*typo*).

3.5. Hasil Pengujian *System Usability Scale* (SUS)

Evaluasi *usability* dilakukan dengan melibatkan 20 responden mahasiswa aktif di wilayah Bogor. Angka 20 ini sangat sejalan dengan jurnal dari Laura Faulkner tahun 2003. Dalam risetnya, Faulkner membuktikan bahwa pengujian dengan 20 pengguna (*sample size of 20*) adalah titik ideal yang mampu mendeteksi hingga 95% sampai 98% masalah kegunaan (*usability*) pada sebuah sistem.

Profil responden: 60% laki-laki dan 40% perempuan; seluruhnya berusia 20–23 tahun; 95% berstatus mahasiswa aktif; dan 95% merupakan pengguna aktif Telegram. Sebesar 95% responden menggunakan teknologi AI seperti ChatGPT secara sering hingga sangat sering dalam kegiatan sehari-hari.

Perhitungan skor SUS dilakukan menggunakan rumus standar John Brooke (1996): untuk pernyataan bernomor ganjil (positif), skor = (Skala Responden − 1); untuk pernyataan bernomor genap (negatif), skor = (5 − Skala Responden); total skor dikalikan 2,5 untuk menghasilkan nilai akhir rentang 0–100 [17]. Berikut rumus yang digunakan dalam perhitungan tersebut:

$$\left(\sum_{n=1}^{5}(\text{Ganjil}_n - 1) + \sum_{n=1}^{5}(5 - \text{Genap}_n)\right) \times 2{,}5$$

Analisis per-butir kuesioner menunjukkan hasil sangat positif: 95% responden menyatakan antarmuka Telegram sangat mudah digunakan dan familiar (P1); 80% responden mengonfirmasi bahwa bot mampu memproses perintah teks dengan akurat (P2); 100% responden menilai respons sistem terasa natural seperti berkomunikasi dengan manusia (P3); 100% responden menilai bot cerdas dalam menangani perintah kompleks (P6); dan 100% responden menolak pernyataan ketidaklayakan sistem sebagai alat bantu kerja sehari-hari (P10). Berikut tabel hasil rekapitulasi perhitungan dengan menggunakan metode SUS:

Responden	Pertanyaan										Total Skor	Skor Akhir $(\times 2,5)$
	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10		
R1	5	1	4	2	5	2	4	2	4	1	34	85
R2	5	1	4	2	4	2	4	2	4	2	32	80
R3	5	3	4	1	4	2	2	4	5	2	28	70
R4	5	1	4	2	4	2	4	2	4	2	32	80
R5	5	2	4	2	4	2	4	2	4	1	32	80
R6	5	5	5	2	4	2	4	2	4	1	30	75
R7	5	2	5	1	4	1	4	2	4	1	35	87,5
R8	5	2	4	1	4	1	4	2	4	1	34	85
R9	5	3	4	3	5	1	4	3	4	2	30	75
R10	5	1	4	1	5	1	5	1	4	1	38	95
R11	5	1	4	1	5	1	5	2	5	1	38	95
R12	5	1	5	1	5	1	5	1	5	1	40	100
R13	5	1	5	1	5	1	5	1	5	1	40	100
R14	5	1	5	1	5	1	5	1	5	1	40	100
R15	5	1	5	1	5	1	5	1	5	1	40	100
R16	5	5	4	1	4	2	5	1	5	1	33	82,5
R17	3	2	5	1	5	2	4	1	5	2	34	85
R18	5	1	5	1	5	1	5	1	5	1	40	100
R19	5	2	5	2	5	1	5	1	5	1	38	95
R20	5	1	5	1	5	1	5	1	5	1	40	100
Rata-rata											**35,4**	**88,5**

Gambar 7. Rekapitulasi Hasil Pengujian

Berdasarkan perhitungan rekapitulasi dari 20 responden, diperoleh rata-rata skor SUS sebesar 88,5. Nilai ini diinterpretasikan menggunakan parameter Bangor *et al*. (2009) [21] sebagai berikut: (1) *Adjective Rating*: skor 88,5 masuk kategori *Excellent*; (2) *Acceptability Ranges*: sistem berada pada zona *Acceptable* (Sangat layak diterima); (3) Grade Scale: sistem memperoleh predikat Grade A. Hasil ini melampaui rata-rata skor SUS industri sebesar 68, membuktikan efektivitas integrasi LLM dan n8n dalam memproses instruksi pengguna.

KESIMPULAN

Berdasarkan hasil perancangan, implementasi, dan pengujian sistem Prioritas Bot, dapat ditarik empat kesimpulan utama. Pertama, arsitektur integrasi Telegram-n8n-LLM terbukti berhasil dibangun, dengan n8n sebagai orkestrator yang menjembatani komunikasi data antar platform secara stabil. Kedua, penerapan GPT-4o mampu memecahkan masalah kekakuan chatbot konvensional dengan memahami instruksi penjadwalan menggunakan gaya bahasa sehari-hari tanpa mengharuskan pengguna menghafal sintaks baku. Ketiga, pemetaan data JSON yang dihasilkan sistem berhasil terintegrasi langsung dengan Google Calendar secara *real-time*, mereduksi beban kognitif pengguna dalam manajemen jadwal manual. Keempat, sistem memperoleh skor SUS rata-rata 88,5 dari 20 responden (kategori

Excellent, Grade A) yang membuktikan kelayakan tinggi sebagai alat bantu produktivitas sehari-hari.

Meskipun demikian, penelitian ini memiliki beberapa keterbatasan yang sekaligus membuka peluang bagi kajian lanjutan. Dari sisi infrastruktur, sistem saat ini masih bergantung pada lingkungan lokal berbasis Docker dan ngrok sehingga ketersediaan layanan (*availability*) tidak dapat dijamin secara penuh; penelitian selanjutnya perlu mengeksplorasi deployment berbasis cloud (misalnya Google Cloud Run atau AWS Lambda) untuk memastikan skalabilitas dan keandalan produksi. Dari sisi antarmuka, integrasi ke platform WhatsApp atau Line dapat memperluas jangkauan pengguna yang tidak menggunakan Telegram sebagai aplikasi pesan utama. Dari sisi kecerdasan, penambahan pangkalan data vektor (*vector database*) seperti Pinecone atau Weaviate berpotensi memberikan memori percakapan jangka panjang yang lebih proaktif dan personal, melampaui batasan konteks sesi yang ada saat ini. Terakhir, dari sisi evaluasi, studi lanjutan dengan desain longitudinal dan sampel yang lebih beragam (lintas profesi dan kelompok usia) akan memperkuat generalisasi temuan serta membuka kajian perbandingan antara model LLM yang berbeda dalam konteks sistem orkestrasi serupa.

DAFTAR PUSTAKA

[1]. R. M. Sinaga, M. A. S. HSB, N. Farezi, F. A. Lubis, dan A. Perdana, "Pengembangan 'LOOPA' Asisten Pengingat Jadwal Pintar Berbasis AI," JATI, vol. 9, no. 4, hlm. 6118–6125, 2025.

[2]. F. Riza, S. J. Al Din, D. Y. Al Afghani, R. Setiabudi, dan Wibisono, "LLM-Based Self-Related Local AI Agent Design through N8N Orchestration for Conversational Memory on RAG," INTECOMS, vol. 8, no. 3, hlm. 939–946, 2025.

[3]. J. Sweller, "Cognitive Load Theory," dalam Psychology of Learning and Motivation, vol. 55, Academic Press, 2011, hlm. 37–76.

[4]. B. K. Britton dan A. Tesser, "Effects of Time-Management Practices on College Grades," Journal of Educational Psychology, vol. 83, no. 3, hlm. 405–410, 1991.

[5]. S. Bellman, R. F. Potter, S. Treleaven-Hassard, J. A. Robinson, dan D. Varan, "The Effectiveness of Branded Mobile Phone Apps," Journal of Interactive Marketing, vol. 25, no. 4, hlm. 191–200, 2011.

[6]. A. Oulasvirta, T. Rattenbury, L. Ma, dan E. Raita, "Habits Make Smartphone Use More Pervasive," Personal and Ubiquitous Computing, vol. 16, no. 1, hlm. 105–114, 2012.

[7]. K. Church dan R. de Oliveira, "What's Up with WhatsApp? Comparing Mobile Instant Messaging Behaviors," dalam Proc. 15th Int. Conf. Human-Computer Interaction with Mobile Devices and Services, 2013, hlm. 352–361.

[8]. A. Bintang, A. L. Hananto, dan A. Hananto, "Telegram Bot Application Development Integration with Google Sheets for Sending Service Reporting," Journal of Artificial Intelligence and Engineering Applications, vol. 4, no. 3, hlm. 2208–2214, 2025.

[9]. J. Achiam *et al.*, "GPT-4 Technical Report," arXiv:2303.08774, 2023.

[10]. Y. Bramanditya dan W. T. Handoko, "Pengembangan Chatbot WhatsApp Menggunakan Otomatisasi AI untuk Meningkatkan Layanan Informasi Pemerintahan," Jutisi, vol. 14, no. 2, hlm. 1195–1206, 2025.

[11]. R. K. Punithavel dan D. Sivakumer, "Large Language Model Framework for Device Orchestration in Low-Code No-Code Solutions," IJCESEN, vol. 11, no. 3, hlm. 5559–5565, 2025.

[12]. A. R. Hevner *et al*., "Design Science in Information Systems Research," MIS Quarterly, vol. 28, no. 1, hlm. 75–105, 2004.

[13]. R. S. Pressman dan B. R. Maxim, Software Engineering: A Practitioner's Approach, 9th ed. New York: McGraw-Hill Education, 2020.

[14]. Sugiyono, Metode Penelitian Pendidikan. Bandung: Alfabeta, 2019.

[15]. D. Merkel, "Docker: Lightweight Linux Containers for Consistent Development and Deployment," Linux Journal, vol. 2014, no. 239, 2014.

[16]. K. Le, Laragon Documentation: A Fast and Lightweight Development Environment for Modern Web Applications. Laragon Official Reference, 2024.

[17]. J. Brooke, "SUS: A 'Quick and Dirty' Usability Scale," dalam Usability Evaluation in Industry, P. W. Jordan et al., Eds. London: Taylor & Francis, 1996, hlm. 189–194.

[18]. R. A. Virzi, "Refining the Test Phase of Usability Evaluation: How Many Subjects is Enough?" Human Factors, vol. 34, no. 4, hlm. 457–468, 1992.

[19]. J. Nielsen, "Why You Only Need to Test with 5 Users," Nielsen Norman Group, 2000. [Online]. Available: https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/. [Diakses: 01-Apr-2026].

[20]. Y. Liu *et al*., "An Empirical Study on Low Code Programming Using Traditional vs Large Language Model Support," arXiv:2402.01156, 2024.

[21]. A. Bangor, P. T. Kortum, dan J. T. Miller, "An Empirical Evaluation of the System Usability Scale," Int. J. Human-Computer Interaction, vol. 24, no. 6, hlm. 574–594, 2008.

[22]. A. Maedche et al., "AI-Based Digital Assistants: Opportunities, Threats, and Research Perspectives," Business & Information Systems Engineering, vol. 61, no. 4, hlm. 535–544, 2019.

[23]. R. López Soto, "Automatización de Procesos con Docker, n8n y Modelos de IA," Tesis de Grado, Univ. Politécnica de Madrid, 2025.