

**PENGEMBANGAN ASISTEN VIRTUAL BERBASIS SUARA
MENGGUNAKAN GENERATIVE AI LARGE LANGUAGE MODEL
UNTUK LAYANAN AKADEMIK UNIVERSITAS MA CHUNG**

TUGAS AKHIR



**UNIVERSITAS
Ma CHUNG**

**Michael Christopher Otniel Wijanto
NIM: 312210015**

**PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI DAN DESAIN
UNIVERSITAS MA CHUNG
MALANG
2025**

LEMBAR PENGESAHAN
TUGAS AKHIR

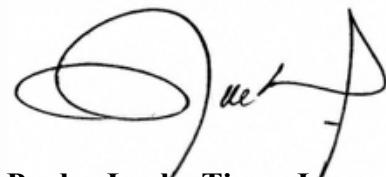
PENGEMBANGAN ASISTEN VIRTUAL BERBASIS SUARA
MENGGUNAKAN GENERATIVE AI LARGE LANGUAGE MODEL
UNTUK LAYANAN AKADEMIK UNIVERSITAS MA CHUNG

Oleh:
Michael Christopher Otniel Wijanto
NIM: 312210015

dari:
PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI DAN DESAIN
UNIVERSITAS MA CHUNG

Telah dinyatakan lulus dalam melaksanakan Tugas Akhir sebagai syarat kelulusan
dan berhak mendapatkan gelar Sarjana Komputer

Dosen Pembimbing 1,



Paulus Lucky Tirma Irawan, S.Kom., MT.
NIP. 20100005

Dosen Pembimbing 2,



Bita Parga Zen, S.Kom., M.Han.
NIP. 20240017

Dekan Fakultas Teknologi dan Desain,



Prof. Dr. Eng. Romy Budhi Widodo
NIP. 20070035

PERNYATAAN KEASLIAN SKRIPSI

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Skripsi saya dengan judul “Pengembangan Asisten Virtual Berbasis Suara Menggunakan Generative AI Large Language Model Untuk Layanan Akademik Universitas Ma Chung” adalah benar benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diizinkan dan bukan merupakan karya pihak lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka. Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Malang, 12 Januari 2026



Michael Christopher Otniel
Wijanto
NIM. 312210015

UNIVERSITAS
MA CHUNG

**PENGEMBANGAN ASISTEN VIRTUAL BERBASIS SUARA
MENGGUNAKAN GENERATIVE AI LARGE LANGUAGE MODEL
UNTUK LAYANAN AKADEMIK UNIVERSITAS MA CHUNG**

Michael Christopher Otniel Wijanto

Universitas Ma Chung

Abstrak

Kemajuan *Artificial Intelligence* (AI) telah mengubah cara manusia berinteraksi dengan teknologi. Namun, asisten virtual generik seperti ChatGPT dan Gemini memiliki keterbatasan dalam konteks layanan institusi pendidikan karena belum terpersonalisasi, tidak dapat dikontrol secara mandiri, dan *knowledge base*-nya tidak dapat dikelola untuk memastikan informasi tetap relevan dan akurat. Penelitian ini mengembangkan asisten virtual berbasis suara menggunakan *Generative AI Large Language Model* untuk layanan akademik Universitas Ma Chung yang dapat dipersonalisasi dan dikelola secara mandiri. Sistem mengintegrasikan *Speech-to-Text* (STT), *Text-to-Speech* (TTS) menggunakan F5-TTS, dan *Retrieval-Augmented Generation* (RAG) dengan Google Gemini 2.5 Flash. Arsitektur *microservice* dibangun menggunakan aplikasi web PHP MVC, n8n yang berjalan di Docker sebagai *orchestrator*, dan Supabase sebagai *vector database*. Sistem dievaluasi oleh 30 responden mahasiswa menggunakan *System Usability Scale* (SUS), Angket Kepuasan, *Mean Opinion Score* (MOS), dan *Word Error Rate* (WER). Hasil evaluasi menunjukkan skor SUS 77,92, Angket Kepuasan 4,21 dari 5, MOS 3,72 dari 5, dan WER 13,13%. Perbandingan akurasi menunjukkan asisten virtual memperoleh skor 86,25%, jauh lebih tinggi dibandingkan Gemini generik yang hanya mencapai 43,12%, membuktikan efektivitas teknik RAG dan *chunking* pengetahuan terkontrol pada *vector database*. Dari sisi latensi, asisten virtual memiliki rata-rata waktu respons 8,24 detik, lebih lambat dibandingkan Gemini dengan 5,1 detik. Sistem berhasil menyediakan layanan informasi akademik yang dapat diakses melalui suara maupun teks dengan *knowledge base* yang terpersonalisasi.

Kata kunci: *asisten virtual, generative AI, speech recognition, text-to-speech, retrieval-augmented generation, microservice*

DEVELOPMENT OF VOICE-BASED VIRTUAL ASSISTANT USING GENERATIVE AI LARGE LANGUAGE MODEL FOR UNIVERSITY MA CHUNG ACADEMIC SERVICES

Michael Christopher Otniel Wijanto

Universitas Ma Chung

Abstract

The advancement of Artificial Intelligence (AI) has transformed how humans interact with technology. However, generic virtual assistants such as ChatGPT and Gemini have limitations in educational institution services as they are not personalized, cannot be independently controlled, and their knowledge bases cannot be managed to ensure information remains relevant and accurate. This research develops a voice-based virtual assistant using Generative AI Large Language Model for Ma Chung University's academic services that can be personalized and independently managed. The system integrates Speech-to-Text (STT), Text-to-Speech (TTS) using F5-TTS, and Retrieval-Augmented Generation (RAG) with Google Gemini 2.5 Flash. The microservice architecture is built using PHP MVC web application, n8n running on Docker as orchestrator, and Supabase as vector database. The system was evaluated by 30 student respondents using System Usability Scale (SUS), Satisfaction Questionnaire, Mean Opinion Score (MOS), and Word Error Rate (WER). Results showed a SUS score of 77.92, Satisfaction score of 4.21 out of 5, MOS of 3.72 out of 5, and WER of 13.13%. Accuracy comparison showed the virtual assistant achieved 86.25%, significantly higher than generic Gemini which only reached 43.12%, proving the effectiveness of RAG techniques and controlled knowledge chunking in vector database. In terms of latency, the virtual assistant had an average response time of 8.24 seconds, slower compared to Gemini's 5.1 seconds. The system successfully provides academic information services accessible through voice or text with a personalized knowledge base.

Keywords: *virtual assistant, generative AI, speech recognition, text-to-speech, retrieval-augmented generation, microservice*

KATA PENGANTAR

Puji syukur, penulis ucapkan terima kasih kepada Tuhan Yang Maha Esa atas berkat dan rahmat-Nya selama pelaksanaan magang. Berkat karunia-Nya, laporan tugas akhir dengan judul “PENGEMBANGAN ASISTEN VIRTUAL BERBASIS SUARA MENGGUNAKAN GENERATIVE AI LARGE LANGUAGE MODEL UNTUK LAYANAN AKADEMIK UNIVERSITAS MA CHUNG” dapat diselesaikan dengan baik.

Dalam penyusunan tugas akhir penulis telah mendapatkan banyak bantuan dari berbagai pihak. Pada kesempatan ini penulis mengucapkan terima kasih kepada:

1. Tuhan yang Maha Esa atas berkat dan karunia-Nya selama proses Praktik Kerja Lapangan hingga laporan ini terselesaikan dengan baik.
2. Paulus Lucky Tirma Irawan, S.Kom., MT. selaku dosen pembimbing 1 yang selalu memberikan bantuan dan dukungan selama proses penggerjaan tugas akhir.
3. Bita Parga Zen, S.Kom., M.Han. selaku dosen pembimbing 2 yang juga selalu memberikan bantuan dan dukungan selama proses penggerjaan tugas akhir.
4. Windra Swastika, S.Kom., MT., Ph.D. selaku ketua dosen pengujii yang sudah memberikan kesempatan kepada penulis untuk menyelesaikan sidang akhir dengan baik.
5. Bapak/Ibu dosen Program Studi Teknik Informatika yang telah memberikan ilmu selama masa studi saya di Universitas Ma Chung
6. Rekan-Rekan Mahasiswa Teknik Informatika Univeristas Ma Chung Angkatan 2022.
7. Universitas Ma Chung Malang yang telah memberikan kesempatan kepada dan fasilitas untuk belajar.

Penulis menyadari bahwa laporan tugas akhir ini masih jauh dari sempurna, oleh karena itu penulis sangat terbuka terhadap kritik yang membangun. Besar harapan penulis agar laporan ini memberikan ilmu dan manfaat bagi semua.

DAFTAR ISI

LEMBAR PENGESAHAN	i
PERNYATAAN KEASLIAN SKRIPSI.....	i
Abstrak.....	ii
Abstract.....	iii
KATA PENGANTAR.....	iv
DAFTAR ISI.....	v
DAFTAR GAMBAR.....	ix
DAFTAR TABEL.....	xii
BAB I.....	1
1.1 Latar belakang	1
1.2 Rumusan Masalah.....	4
1.3 Batasan Masalah	5
1.4 Tujuan Penelitian	5
1.5 Manfaat Penelitian	6
1.6 Rencana Penelitian.....	7
1.7 Luaran.....	8
BAB II	9
2.1 Assisten Virtual	9
2.1.1 Definisi dan Konsep Asisten Virtual	9
2.1.2 Sejarah dan Evolusi Asisten Virtual	9
2.1.3 Jenis-jenis Asisten Virtual	10
2.1.4 Implementasi Asisten Virtual dalam Layanan Akademik Universitas Ma Chung	11
2.2 Kecerdasan Buatan dan Large Language Model.....	12
2.2.1 Artificial Intelligence dalam Pendidikan Tinggi	12
2.2.2 Large Language Model (LLM)	13
2.2.3 Google Gemini 2.5 Flash.....	14
2.3 Generative AI dan Natural Language Processing	15
2.3.1 Generative AI	15
2.3.2 Natural Language Processing (NLP).....	15

2.4	Teknologi Speech Processing	16
2.4.1	Speech-to-Text (STT).....	16
2.4.2	Text-to-Speech (TTS).....	17
2.4.3	F5-TTS	18
2.4.4	Chatterbox	19
2.5	Retrieval-Augmented Generation (RAG).....	20
2.5.1	Konsep dan Arsitektur RAG.....	20
2.5.2	Vector Database dan Embedding	21
2.5.3	Integration Framework	22
2.6	Sistem Informasi Akademik.....	23
2.6.1	Karakteristik Layanan Akademik Universitas	23
2.6.2	Digitalisasi Layanan Akademik	24
2.7	Arsitektur dan Integrasi Sistem.....	25
2.7.1	Arsitektur Microservices	25
2.7.2	Web Application Framework.....	26
2.7.3	Integration Patterns	28
2.8	Evaluasi dan Metrik Kinerja	29
2.8.1	Metrik Akurasi Sistem.....	29
2.8.2	User Experience Metrics.....	30
2.9	Blackbox Testing	32
2.10	Penelitian Terkait dan State-of-the-Art	32
2.11	Kerangka Teoritis Penelitian	34
2.11.1	Technology Acceptance Model (TAM).....	34
2.11.2	Human-Computer Interaction (HCI).....	34
BAB III		36
3.1	Alur Penelitian.....	36
3.2	Analisis Kebutuhan Sistem	37
3.3	Perancangan Alur Sistem Asisten Virtual	40
3.4	Arsitektur Web Application.....	41
3.5	Perancangan Arsitektur LLM dengan n8n	42
3.5.1	Implementasi n8n pada Docker	42
3.5.2	Perancangan Workflow n8n.....	43

3.5.3	Integrasi LLM Gemini 2.5 Flash Pada n8n	44
3.5.4	Implementasi Retrieval-Augmented Generation (RAG).....	44
3.5.5	Implementasi Vector Database	45
3.5.6	Pemberian Knowledge Base pada Model.....	45
3.5.7	Konfigurasi Webhook.....	46
3.6	Perancangan Arsitektur Text-to-Speech (F5-TTS).....	46
3.6.1	Implementasi F5-TTS dan Chatterbox Menggunakan Virtual Environment Python.....	46
3.6.2	Penyediaan Endpoint F5-TTS dan Chatterbox.....	47
3.6.3	Konfigurasi Referensi Audio dan Referensi Teks.....	47
3.6.4	Optimasi Generatif Audio	48
3.7	Implementasi Speech-to-Text (STT)	48
3.8	Integrasi Arsitektur Microservice	49
3.8.1	Integrasi Web Application dengan n8n	49
3.8.2	Integrasi Web Application dengan F5-TTS.....	49
3.9	Mekanisme Fallback	50
3.10	Pengujian dan Evaluasi Rancangan Sistem.....	50
3.10.1	Pengujian dan Evaluasi Non-Fungsionalitas	51
3.10.2	Pengujian dan Evaluasi Hasil Generatif AI	52
3.10.3	Pengujian dan Evaluasi Hasil Suara TTS.....	53
3.10.4	Pengujian dan Evaluasi Hasil Text STT	54
3.10.5	Pengujian Perbandingan Akurasi dan Latensi Asisten Virtual dengan Gemini.....	54
BAB IV	56
4.1	Gambaran Umum dan Alur Web Application Asisten Virtual	56
4.2	Tuning Knowledge Base	57
4.3	Persiapan Penelitian.....	58
4.3.1	Persiapan Perangkat.....	58
4.3.2	Persiapan Software	59
4.3.3	Persiapan Web Application dengan PHP MVC.....	59
4.3.4	Persiapan Database MySQL	60
4.3.5	Persiapan API Gemini	62

4.3.6	Persiapan Vektor Database Supabase.....	62
4.3.7	Persiapan Reranker	63
4.3.8	Persiapan N8N.....	64
4.3.9	Persiapan F5_TTS.....	68
4.4	Hasil Web Application.....	69
4.4.1	Halaman Login	69
4.4.2	Navigasi dan Header	71
4.4.3	Halaman Dashboard Admin	72
4.4.4	Halaman Knowledge Base Admin	75
4.4.5	Halaman Virtual Assistant	79
4.5	Hasil N8N Workflow.....	82
4.5.1	Main Virtual Assistant Workflow	82
4.5.2	Upload Knowledge Base Workflow.....	86
4.5.3	Workflow Analitik Total Riwayat Percakapan.....	87
4.6	Hasil TTS dengan F5_TTS.....	88
4.7	Contoh Pertanyaan yang Digunakan Pada Waktu Evaluasi	89
4.8	Contoh Percakapan dengan Asisten Virtual Pada Saat Evaluasi ..	90
4.9	Hasil Evaluasi System Usability Scale.....	92
4.10	Hasil Evaluasi Angket Kepuasan.....	92
4.11	Hasil Evaluasi Mean Opinion Score.....	93
4.12	Hasil Evaluasi Word Error Rate	94
4.13	Hasil Evaluasi Blackbox	95
4.14	Hasil Perbandingan Teknik Chunking RAG dan Implementasi Supabase Pada Virtual Asisten dengan Gemini Generik.....	96
4.15	Hasil Pengujian Pertanyaan Tidak Relevan.....	99
4.16	Solusi Keterbatasan TTS Sementara	101
4.17	Github Repository.....	102
BAB V.....		103
5.1	Kesimpulan	103
5.2	Saran.....	103
Daftar Pustaka.....		105
Lampiran		110

DAFTAR GAMBAR

Gambar 3.1 Alur Penelitian.....	36
Gambar 3.2 Alur Sistem Asisten Virtual	41
Gambar 3.3 Rancangan Workflow n8n.....	43
Gambar 4.1 Tuning Dokumen <i>Knowlegde Base</i>	58
Gambar 4.2 Struktur <i>Framework PHP MVC</i>	60
Gambar 4.3 Tabel Database Web Applicaton.....	61
Gambar 4.4 Konfigurasi Koneksi Database.....	62
Gambar 4.5 Google API Studio	62
Gambar 4.6 Tabel Vektor Database Supabase	63
Gambar 4.7 API Cohere Reranker	64
Gambar 4.8 Docker <i>Container</i> n8n.....	64
Gambar 4.9 <i>Setting Google Gemini Node</i>	65
Gambar 4.10 <i>Setting Kredensial Gemini</i>	65
Gambar 4.11 <i>Setting Node</i> Supabase	66
Gambar 4.12 <i>Setting Kredensial Supabase</i>	66
Gambar 4.13 Setting <i>Node Postgres Chat Memory</i>	67
Gambar 4.14 Setting Kredensial <i>Node Postgres Chat Memory</i>	67
Gambar 4.15 <i>Setting Node</i> Reranker.....	68
Gambar 4.16 Setting Kredensial Reranker.....	68
Gambar 4.17 Halaman Login Web Application	69
Gambar 4.18 Spesifikasi Database <i>User Management</i>	70
Gambar 4.19 <i>Sidebar Navigasi Admin</i>	71
Gambar 4.20 Header Admin	72
Gambar 4.21 Header Guest.....	72
Gambar 4.22 Halaman <i>Dashboard Admin</i>	72
Gambar 4.23 Database Fitur Analitik Pertanyaan.....	74
Gambar 4.24 Halaman List Dokumen Knowledge Base Admin	75
Gambar 4.25 Notifikasi Sistem Pemberitahuan	76
Gambar 4.26 Halaman Unggah Dokumen	77

Gambar 4.27 Halaman Berhasil Unggah Dokumen.....	77
Gambar 4.28 Penyimpanan Data <i>Knowlede</i> pada Supabase	78
Gambar 4.29 Spesifikasi Database <i>Document Management</i>	79
Gambar 4.30 Halaman Asisten Virtual Mode Suara.....	80
Gambar 4.31 Halaman Asisten Virtual Mode Suara Ketika Service TTS Tidak Berfungsi	80
Gambar 4.32 Halaman Asosten Virtual Mode Suara Ketika Service n8n Tidak Berfungsi	81
Gambar 4.33 Halaman Asisten Virtual Mode Chat	81
Gambar 4.34 Halaman Asisten Virtual Mode Chat Ketika Service n8n Tidak Berfungsi	82
Gambar 4.35 <i>Workflow</i> n8n Utama Asisten Virtual	82
Gambar 4.36 Konfigurasi <i>promt node AI Agent</i>	83
Gambar 4.37 <i>Workflow</i> Fitur Analitik Jenis Pertanyaan.....	84
Gambar 4.38 Konfigurasi <i>promt node AI Agent</i> Analitik	85
Gambar 4.39 <i>Workflow</i> <i>Clear Knowledge Base Document</i>	86
Gambar 4.40 <i>Workflow</i> <i>Upload Knowledge Base Document</i>	86
Gambar 4.41 <i>Workflow</i> Analitik Total Riwayat Percakapan.....	87
Gambar 4.42 Potongan <i>Code Endpoint</i> <i>WebSocket</i> Service TTS	89
Gambar 4.43 Contoh Jawaban yang Diharapkan.....	91
Gambar 4.44 Contoh Jawaban yang Tidak Diharapkan.....	91
Gambar 4.45 Grafik Skor SUS.....	92
Gambar 4.46 Grafik Skor Likert	92
Gambar 4.47 Grafik Skor MOS	93
Gambar 4.48 Grafik Skor WER	94
Gambar 4.49 Grafik Skor Perbandingan Akurasi Jawaban Asisten Virtual dengan Gemini.....	97
Gambar 4.49 Grafik Skor Perbandingan Rata-rata Latensi Jawaban Asisten Virtual dengan Gemini	98
Gambar 4.49 Jawaban Gemini Tentang Spesifikasi Server Gemini	99
Gambar 4.50 Harga Penawaran ElevenLab	102

DAFTAR TABEL

Tabel 3.1 Komponen Perangkat Keras3;	38
Tabel 3.2 Komponen Perangkat Keras	38
Tabel 3.3 Komponen Perangkat Lunak	39
Tabel 3.4 Tabel Pernyataan <i>System Usability Scale</i>	52
Tabel 3.5 Pertanyaan <i>Likert Scale Evaluation</i>	53
Tabel 3.6 Pernyataan <i>Mean Opinion Score</i>	53
Tabel 4.1 Contoh Pertanyaan Saat Evaluasi	89
Tabel 4.2 Blackbox	95
Tabel 4.3 Poin Akurasi Jawaban Asisten Virtual	96
Tabel 4.4 Poin Rata-rata Latensi Jawaban Asisten Virtual	96
Tabel 4.5 Poin Akurasi Jawaban Gemini	96
Tabel 4.6 Poin Rata-rata Latensi Jawaban Gemini	97
Tabel 4.7 Pengujian Jawaban Tidak Relevan	99

UNIVERSITAS
MA CHUNG

BAB I

Pendahuluan

1.1 Latar belakang

Kemajuan *Artificial Intelligence* (AI) secara mendasar telah mengubah cara kerja, belajar, dan berinteraksi manusia dengan komputer. Transformasi digital saat ini ditandai oleh penerapan AI yang luas di berbagai sektor, menghasilkan peningkatan efisiensi dan produktivitas yang belum pernah terjadi sebelumnya. Laporan Microsoft *Work Trend Index* 2024 menunjukkan bahwa 75% pekerja pengetahuan global kini memanfaatkan AI, di mana 90% melaporkan penghematan waktu, 85% fokus pada tugas krusial, dan 84% merasa lebih kreatif (Microsoft, 2024).

Perkembangan *Large Language Model* (LLM) seperti ChatGPT, Google Gemini, dan Claude, sebagai model AI, telah mengubah signifikan cara kita mengakses dan memproses informasi. Asisten virtual AI ini kini menjadi bagian penting dari aktivitas harian, membantu pengguna menyelesaikan berbagai tugas mulai dari penulisan, analisis data, hingga pengambilan keputusan secara instan dan ringkas.

Dalam dunia teknologi, misalnya, muncul konsep "*vibe coding*" yang merevolusi kerja *programmer*. Dulu para *programmer* mencari manual ilmu dan refrensi dari buku tebal, dokumentasi, atau forum seperti Stack Overflow, namun saat ini sudah dapat dilakukan *coding* dengan bantuan AI yang memahami konteks dan memberikan solusi *real-time* (Gadde A., 2025). Salamun et al (2024) menegaskan bahwa optimasi *chatbot* dengan pemanfaatan *Natural Language Processing* telah membuka dimensi baru dalam interaksi manusia-mesin, memungkinkan sistem untuk memahami konteks percakapan yang kompleks dan memberikan respons yang relevan secara cepat.

Meskipun asisten virtual AI generik seperti ChatGPT, Gemini, dan Claude menawarkan kemampuan yang luar biasa, implementasinya dalam konteks organisasi dan institusi menghadapi sejumlah keterbatasan fundamental yang signifikan. Pertama, asisten virtual publik ini tidak memiliki personalisasi informasi yang dibutuhkan oleh institusi spesifik (Cong-Lem N. et al., 2024). Kedua, sistem ini tidak dapat melakukan pembaruan data secara *real-time* dan terkontrol sesuai

dengan kebutuhan organisasi yang dinamis (Weber M. et al., 2022). Ketiga, aspek keamanan dan privasi data menjadi perhatian serius karena informasi sensitif tidak mungkin diekspos ke sistem eksternal yang tidak dikontrol langsung oleh institusi (Chen Y. & Esmaeilzadeh P., 2024). Keempat, keterbatasan interaksi yang hanya berbasis teks membuat pengguna harus menghentikan aktivitas utama mereka untuk mengetik pertanyaan, sehingga tidak memungkinkan *multitasking* yang efektif dan pengalaman *hands-free* yang sesungguhnya dibutuhkan dalam lingkungan kerja yang dinamis (Tan J. T. et al., 2025)

Keterbatasan-keterbatasan ini menghadirkan tantangan khusus dalam konteks pendidikan tinggi, dimana kebutuhan akan informasi akademik yang akurat, terkini, spesifik, dan mudah diakses. Seperti yang diungkapkan oleh Guntoro, Costaner, & Lisnawita (2020), *chatbot* berbasis *Artificial Intelligence Markup Language* (AIML) mampu memberikan layanan informasi dan akademik kampus dengan respons otomatis yang memudahkan mahasiswa dalam mengakses berbagai informasi terkait registrasi, program studi, hingga prosedur akademik lainnya. Namun, pendekatan AIML ini memiliki keterbatasan dalam fleksibilitas dan kemampuan pemahaman bahasa natural yang kompleks, terutama ketika menghadapi pertanyaan yang bervariasi dan membutuhkan pemahaman konteks yang mendalam.

Pendekatan *Natural Language Processing* (NLP) terbukti penting dalam membangun *chatbot* layanan akademik yang mampu menanggapi pertanyaan mahasiswa secara fleksibel dan efisien. Penelitian oleh Mulyatun et al. (2021) tentang *chatbot* informasi akademik menggunakan pendekatan NLP dengan metode *Fuzzy String Matching*, menegaskan bahwa teknologi ini efektif dalam mengatasi keterbatasan staf administrasi dan menyediakan layanan informasi 24 jam. Studi ini menunjukkan bahwa dengan NLP, *chatbot* dapat merespons setiap pertanyaan mahasiswa tanpa batasan waktu, yang sangat membantu mengatasi kesalahpahaman informasi yang sering terjadi. Penemuan ini memperkuat argumen bahwa chatbot AI dapat meningkatkan efisiensi dan kualitas layanan, meskipun tantangan tetap ada dalam memastikan basis pengetahuan yang komprehensif untuk setiap variasi bahasa alami yang digunakan mahasiswa.

Berlanjut pada perkembangan teknologi *Retrieval-Augmented Generation* (RAG) menawarkan solusi inovatif untuk mengatasi keterbatasan asisten virtual generik. RAG merupakan paradigma baru dalam AI yang mengintegrasikan kemampuan generasi bahasa dari LLM dengan sistem *retrieval* informasi dari *knowledge base* yang terkontrol dan dapat diperbarui secara dinamis (Lewis et al., 2020). Teknologi ini memungkinkan asisten virtual untuk memberikan respons yang tidak hanya berdasarkan pengetahuan umum model, tetapi juga informasi spesifik dan terkini dari database, sehingga menghasilkan jawaban yang lebih akurat, relevan, dan terpercaya. Pratama, & Sisepaputra (2025) dalam pengembangan sistem *helpdesk* kampus menggunakan RAG mendemonstrasikan bahwa dengan mengintegrasikan model Gemini dan pencarian semantik (FAISS), sistem berhasil memberikan respons yang cepat, akurat, dan andal, menyederhanakan interaksi pengguna dengan layanan administratif kampus.

Integrasi teknologi suara sebagai antarmuka utama memberikan nilai tambah yang signifikan dalam konteks layanan informasi akademik. Asisten virtual pada dasarnya merupakan serangkaian bahasa pemrograman berbasis NLP yang memungkinkan pengguna untuk berkomunikasi dan mendapatkan respon dari aplikasi dengan cara yang sama seperti berkomunikasi dengan individu lain (Perdana & Irwansyah, 2019). Teknologi berbasis suara ini memungkinkan mahasiswa untuk mengakses informasi akademik sambil melakukan aktivitas lain, menciptakan pengalaman *multitasking* yang efisien dan natural. *Interaksi hands-free* ini sangat relevan dengan tren modern dimana efisiensi waktu dan kemudahan akses menjadi prioritas utama pengguna teknologi.

Universitas Ma Chung sebagai institusi pendidikan tinggi yang progresif dan berkomitmen pada inovasi teknologi menghadapi tantangan serupa dalam penyampaian layanan informasi akademik. Sistem layanan tradisional yang terbatas oleh jam operasional kantor, ketersediaan staf yang menginginkan akses informasi secara fleksibel dengan respons yang cepat dan akurat. Informasi akademik yang tersebar di berbagai unit administratif, website, dan buku panduan “*Student Guide*” yang tebal membuat sulit dalam proses pencarian informasi, mengakibatkan mahasiswa menghabiskan waktu yang tidak efisien untuk mendapatkan jawaban atas pertanyaan-pertanyaan akademik mereka.

Dari situ pengembangan asisten virtual berbasis suara dengan teknologi RAG menggunakan *Generative AI Large Language Model* menawarkan solusi komprehensif untuk mengatasi permasalahan kompleks tersebut. Sistem ini mengintegrasikan empat komponen teknologi utama, yaitu LLM untuk pemahaman dan generasi bahasa natural yang canggih, arsitektur RAG untuk *retrieval* informasi spesifik dari *knowledge base* Universitas Ma Chung yang selalu terkini dan terkontrol, teknologi *Speech-to-Text* dan *Text-to-Speech* untuk interaksi berbasis suara yang natural dan *hands-free*, serta sistem manajemen *knowledge base* untuk memastikan akurasi, relevansi, dan keamanan informasi akademik.

Selain itu terdapat keunggulan utama sistem yang dikembangkan terletak pada personalisasi dan spesialisasi yang tinggi terhadap konteks Universitas Ma Chung serta penjagaan informasi yang privasi dari Universitas Ma Chung. Berbeda dengan asisten virtual generik yang memberikan jawaban umum, sistem ini dirancang khusus untuk memahami struktur universitas, kebijakan akademik, prosedur administratif, dan karakteristik unik dari Universitas Ma Chung. *Knowledge base* seperti “*Student Guide*” yang terkelola secara internal memastikan bahwa setiap informasi yang diberikan telah diverifikasi dan sesuai dengan regulasi terkini universitas, mengeliminasi risiko informasi yang keliru atau *outdated* yang sering terjadi pada asisten virtual publik.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan, maka rumusan masalah dalam penelitian ini adalah:

1. Bagaimana mengimplementasikan *Generative AI Large Language Model* untuk memberikan informasi yang jelas dan akurat dengan pengujian menggunakan Angket Kepuasan dengan skor ≥ 3 .
2. Bagaimana mengintegrasikan teknologi *Speech-to-Text* (STT) untuk memudahkan interaksi *hand-free* untuk pengguna dengan tolak ukur pengujian menggunakan matriks *word error rate* (WER) $\leq 20\%$..
3. Bagaimana mengintegrasikan teknologi *Text-to-Speech* (TTS) untuk meningkatkan efisiensi komunikasi asisten virtual dengan tolak ukur pengujian menggunakan matriks *Mean Opinion Score* (MOS) ≥ 3

4. Bagaimana memastikan asisten virtual dapat membantu mahasiswa Universitas Ma Chung dalam mencari informasi secara mudah, efisien dan akurat melalui pengujian *System Usability Scale* (SUS) ≥ 60 atau *acceptable*.

1.3 Batasan Masalah

Berdasarkan rumusan masalah yang telah diuraikan, maka ditetapkan batasan masalah dalam penelitian ini sebagai berikut:

1. *User interface* akan dibatasi dengan menggunakan web *application* yang dibangun menggunakan PHP MVC.
2. Model AI untuk LLM akan menggunakan Google Gemini 2.5 Flash dan terbatas pada jumlah *token* yang disediakan pada layanan tidak berbayar.
3. Pemberian *knowlage* pada AI akan dibatasi dengan penggunaan Supabase sebagai *database* vektor.
4. Untuk menghubungkan LLM dengan RAG dibatasi dengan penggunaan n8n yang dijalankan menggunakan docker.
5. Teknologi TTS dibatasi dengan menggunakan *open source code* pihak ketiga F5-TTS yang terbatas hanya memiliki kemampuan optimal untuk bahasa inggris dan mandarin saja.
6. Teknologi STT dibatasi dengan menggunakan API Browser modern yang dijalankan dengan JavaScript.
7. Web *application* akan dijalankan secara lokal karena tingkat kerumitan dan biaya untuk melakukan *hosting* cukup tinggi.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah:

1. Mengimplementasikan *Generative AI Large Language Model* untuk memproses informasi akademik dan memberikan respons yang akurat melalui interaksi suara.
2. Mengintegrasikan teknologi *Text-to-Speech* (TTS) dan *Speech-to-Text* (STT) untuk mendukung komunikasi natural berbasis suara antara pengguna dan sistem.
3. Mengevaluasi kinerja sistem asisten virtual yang dikembangkan dari aspek akurasi respons, kecepatan respons, dan kepuasan pengguna.

4. Menghasilkan asisten virtual yang dapat dipersonalisasi dan dikontrol sendiri, membuatnya unik berbeda dengan asisten virtual pada umumnya.

1.5 Manfaat Penelitian

Penelitian ini diharapkan memberikan manfaat sebagai berikut:

a. Bagi Universitas

1. Memberikan kontribusi dalam pengembangan layanan akademik dengan memanfaatkan teknologi AI yang dapat memberikan layanan informasi kepada mahasiswa tanpa harus terikat bertanya kepada staf dengan tetap menjamin keakuratan jawaban.
2. Menjadi studi kasus dan referensi berharga untuk implementasi teknologi *Generative AI*, *Retrieval-Augmented Generation* (RAG), dan arsitektur *microservice* dalam lingkungan kampus, mendorong inovasi digital dan menjadi *benchmark* bagi pengembangan sistem serupa di masa depan.
3. Dengan asisten virtual yang dapat dikontrol secara mandiri, dapat membantu pihak universitas memantau kebutuhan mahasiswa terhadap layanan / fasilitas Universitas melalui interaksi asisten virtual.

b. Bagi Mahasiswa

1. Menyediakan akses informasi akademik kampus yang lebih cepat, *real-time*, dan intuitif melalui interaksi suara yang natural, meningkatkan kualitas pengalaman administrasi mahasiswa tanpa perlu mencari dokumen secara manual.
2. Memperoleh kesempatan untuk berkembang melalui penelitian yang dilakukan, mengaplikasikan pengetahuan teknis dalam membangun solusi AI yang kompleks dan *end-to-end* yang memiliki dampak nyata.
3. Menyediakan metode interaksi yang fleksibel berupa teks dan suara, yang dapat mengakomodasi preferensi pengguna yang berbeda dan berpotensi meningkatkan keterlibatan pengguna dengan layanan informasi kampus.

1.6 Rencana Penelitian

Tabel 1.1 Rencana Penelitian

Proses	September				Oktober				November				Desember			
	w1	w2	w3	w4	w1	w2	w3	w4	w1	w2	w3	w4	w5	w1	w2	w3
Studi Liteatur																
Eksperimen																
Pengujian dan Evaluasi																
Pelaporan																
Sidang Akhir																

Penelitian ini dijadwalkan berjalan sekitar empat bulan, dari awal September hingga minggu ketiga Desember. Semua tahapan penelitian diatur secara terstruktur untuk menjamin kesinambungan dan kelancaran proses. Kegiatan dimulai dengan Studi Literatur, yang berlangsung selama bulan September penuh, berfokus pada penguatan landasan teoretis, pemahaman teknologi *Generative AI LLM*, cara mengintegrasikan n8n, memilih teknologi TTS yang cocok dengan permasalahan dan arsitektur *microservice* yang dapat menghubungkan semua sistem. Selain itu pada tahapan ini juga mencari tahu lebih detail tentang dokumen resmi Universitas Ma Chung secara spesifik “*Student Guide*” yang akan dijadikan *knowlage base* dan dilakukan analisis kebutuhan yang dapat diakomodasi pada asisten virtual

. Paralel dengan studi literatur, Eksperimen dimulai pada pertengahan September dan berlanjut hingga akhir November. Fase eksperimen ini meliputi desain arsitektur sistem, implementasi *workflow* n8n, *integrasi* Gemini 2.5 Flash, dan pengembangan komponen STT serta TTS. Memasuki pertengahan November hingga minggu kedua Desember, fokus penelitian beralih ke Pengujian dan Evaluasi mendalam terhadap hasil akhir asisten virtual. Tahap ini krusial untuk memverifikasi akurasi respons, kualitas audio, kinerja integrasi *microservice*, dan efektivitas pembatasan fungsi sistem. Secara berkelanjutan, proses pelaporan progres penelitian dilakukan secara terus-menerus sejak September hingga Desember. Seluruh rangkaian penelitian akan ditutup dengan Sidang Akhir pada minggu ketiga Desember.

1.7 Luaran

Luaran utama dari penelitian ini adalah sistem Asisten Virtual berbasis suara yang berfungsi secara *end-to-end*, dirancang khusus untuk layanan informasi akademik dan administratif Universitas Ma Chung. Sistem ini merupakan implementasi nyata dari arsitektur *microservice* yang mengintegrasikan *Generative AI LLM* (Gemini 2.5 Flash) dengan mekanisme *Retrieval-Augmented Generation* (RAG) melalui *workflow* n8n dan *vector database* Supabase. Sistem ini dapat dilakukan interaksi langsung dengan adanya *User Interface* dari web *application* dengan semua asset pendukung yang akan menjadi daya tarik pertama dari asisten virtual.

Selain luaran berupa produk sistem, penelitian ini juga menghasilkan luaran akademik berupa artikel ilmiah publikasi yang memuat hasil evaluasi kinerja sistem secara terperinci. Publikasi ini akan mencakup metodologi implementasi teknologi STT/TTS, arsitektur integrasi *microservice*, serta hasil dari evaluasi dengan beberapa metrik yang mengukur akurasi, kecepatan dan kualitas respons. Luaran ini diharapkan dapat menjadi studi kasus dan referensi berharga mengenai implementasi *Generative AI* dalam konteks layanan akademik kampus di Indonesia.

UNIVERSITAS
MA CHUNG

BAB II

Tinjauan Pustaka

2.1 Assisten Virtual

2.1.1 Definisi dan Konsep Asisten Virtual

Asisten virtual merupakan teknologi yang menggabungkan kecerdasan buatan dengan kemampuan interaksi natural untuk memberikan layanan otomatis kepada pengguna. Asisten virtual pada dasarnya merupakan serangkaian bahasa pemrograman dengan basis NLP (*Natural Language Program*) yang memungkinkan pengguna untuk berbicara dan mendapatkan respon berupa balasan dari aplikasi dengan cara yang sama halnya dengan apa yang terjadi pada individu-individu lainnya. Teknologi ini memungkinkan pengguna untuk mengakses informasi secara *real-time* melalui berbagai platform digital yang terintegrasi. Asisten virtual menggabungkan algoritma pembelajaran mesin dengan basis pengetahuan yang terstruktur untuk menghasilkan respons yang akurat dan kontekstual. Kemampuan adaptasi dan personalisasi menjadi karakteristik utama yang membedakan asisten virtual dari sistem informasi konvensional.

2.1.2 Sejarah dan Evolusi Asisten Virtual

Sejarah asisten virtual berakar jauh sebelum era perangkat pintar modern, bermula dari upaya awal dalam bidang kecerdasan buatan (AI) dan pemrosesan bahasa alami (NLP). Salah satu tonggak sejarah paling signifikan adalah pengembangan ELIZA pada pertengahan tahun 1960-an oleh Joseph Weizenbaum di MIT. ELIZA adalah program komputer awal yang mampu melakukan percakapan dasar dengan mengidentifikasi kata kunci dan menggunakan pola skrip untuk merespons, meniru seorang terapis Rogerian. Meskipun fungsionalitasnya terbatas dan tidak memiliki pemahaman sejati, ELIZA meletakkan dasar konseptual untuk interaksi manusia-komputer menggunakan bahasa alami. Perkembangan selanjutnya pada tahun 1970-an melibatkan proyek seperti HARPY di Carnegie Mellon University, yang fokus pada pengenalan ucapan berskala besar, menunjukkan bahwa mesin dapat mulai memahami dan memproses perintah suara secara lebih kompleks.

Evolusi asisten virtual mulai memasuki fase komersial dan praktis pada akhir abad ke-20 dan awal abad ke-21. Kemajuan signifikan dalam teknologi pengenalan suara dan komputasi yang lebih cepat memungkinkan lahirnya asisten virtual yang terintegrasi. Contohnya adalah Microsoft Office Assistant (Clippy) yang muncul pada tahun 1997, meskipun terbatas pada tugas dalam aplikasi, menandai langkah awal integrasi asisten dalam perangkat lunak sehari-hari. Titik balik krusial terjadi dengan munculnya Siri yang diperkenalkan pada iPhone 4S pada tahun 2011, yang merupakan aplikasi pertama yang menggabungkan pengenalan suara, NLP, dan layanan web untuk menjalankan tugas yang lebih luas. Kesuksesan Siri mendorong perusahaan teknologi lain untuk berinvestasi besar-besaran, menghasilkan Google Assistant, Amazon Alexa, dan Microsoft Cortana, yang masing-masing menggunakan *machine learning* dan *deep learning* untuk meningkatkan akurasi dan konteks percakapan, mengubah asisten virtual dari sekadar alat menjadi bagian integral dari ekosistem digital (Coheur, 2020).

Saat ini, asisten virtual terus berkembang menuju kemampuan yang lebih canggih, didorong oleh kemajuan dalam model AI generatif dan kemampuan untuk berinteraksi dengan internet untuk segala (IoT). Asisten modern tidak hanya merespons perintah, tetapi juga memprediksi kebutuhan pengguna, mengelola tugas yang kompleks, dan mempertahankan konteks dalam percakapan yang panjang. Peningkatan kemampuan ini juga memicu munculnya isu-isu penting terkait etika AI, privasi data, dan keandalan sistem, yang menjadi fokus utama dalam penelitian kontemporer. Para peneliti saat ini sedang mengeksplorasi cara untuk membuat asisten virtual menjadi lebih proaktif, mampu berempati, dan dapat melakukan penalaran layaknya manusia, yang mana menunjukkan masa depan di mana antarmuka berbasis suara menjadi cara dominan untuk berinteraksi dengan teknologi.

2.1.3 Jenis-jenis Asisten Virtual

Klasifikasi asisten virtual mencerminkan keragaman fungsionalitas dan domain aplikasi yang terus berkembang, melampaui sekadar perangkat konsumen. Pembagian utama dapat dilihat berdasarkan target pengguna dan fokus operasionalnya, yang meliputi Asisten Suara Pribadi (*Personal Virtual Assistants*) seperti Siri dan Alexa yang membantu tugas harian, hingga Asisten Bisnis

(*Business Virtual Assistants*) seperti IBM Watson yang dirancang untuk mendukung operasional perusahaan. Kategorisasi yang lebih rinci juga mencakup Asisten Khusus Domain, seperti Asisten Kesehatan untuk manajemen gejala dan janji temu, dan Asisten Pendidikan untuk mendukung proses belajar mengajar, di mana setiap jenisnya menggunakan algoritma AI untuk memahami konteks spesifik bidangnya, Okpomu, E. B, (2024). Evolusi ini menunjukkan bahwa asisten virtual telah menjadi alat strategis yang vital di sektor-sektor profesional, tidak hanya berfungsi sebagai alat bantu hiburan atau informasi dasar.

Jenis-jenis asisten virtual juga dapat diklasifikasikan berdasarkan mekanisme interaksi dan teknologi inti yang digunakan, yaitu antara Asisten Berbasis Suara dan Asisten Berbasis Teks (*Chatbots*). Asisten Berbasis Suara yang sering disebut *Virtual Personal Assistants* (VPA) mengandalkan teknologi *Automatic Speech Recognition* (ASR) untuk mengubah input suara menjadi teks, yang kemudian diproses oleh *Natural Language Processing* (NLP) untuk memahami maksud perintah. Penggunaan VPA ini telah didorong oleh kemudahan fitur *hands-free* pada perangkat *smart device* (Jaber et al., 2024). Sementara itu, asisten berbasis teks atau *chatbot* berkisar dari sistem berbasis aturan sederhana untuk layanan pelanggan hingga sistem yang didukung *Large Language Models* (LLMs) yang mampu berinteraksi secara kompleks dan menghasilkan respons yang kontekstual, menawarkan skalabilitas tinggi dalam lingkungan digital.

2.1.4 Implementasi Asisten Virtual dalam Layanan Akademik Universitas Ma Chung

Penerapan asisten virtual di bidang akademik adalah salah satu solusi bagi institusi pendidikan tinggi untuk mengoptimalkan efisiensi penyediaan layanan informasi kepada mahasiswa dan seluruh civitas akademika. Pengembangan prototipe asisten virtual berfungsi sebagai pusat layanan informasi mahasiswa yang mampu beroperasi selama 24 jam tanpa batasan waktu. Model layanan yang otomatis ini secara signifikan mengatasi keterbatasan layanan konvensional yang sangat bergantung pada jam kerja dan kehadiran fisik staf administratif. Asisten virtual akademik ini dirancang untuk memberikan respons otomatis terhadap berbagai pertanyaan umum mahasiswa, dengan tetap menjamin keakuratan dari jawaban. Di Universitas Ma Chung sering kali mahasiswa memerlukan untuk

pencarian informasi, namun informasi yang hendak dicari sangat mungkin tersebar di beberapa tempat. Informasi lengkap yang tersedia pada “*Studen Guide*” juga menghambat kecepatan mahasiswa untuk memperoleh informasi karena harus mencari secara manual dari buku yang tebal. Dengan bantuan asisten virtual, semua informasi dapat dikumpulkan menjadi satu *knowlage base* dan mahasiswa cukup bertanya pada asisten virtual selayaknya bertanya kepada staf akademik.

2.2 Kecerdasan Buatan dan Large Language Model

2.2.1 Artificial Intelligence dalam Pendidikan Tinggi

Artificial Intelligence (AI) telah membawa transformasi mendasar dalam sistem pendidikan tinggi, terutama melalui modernisasi dan peningkatan efisiensi layanan akademik. Implementasi AI memungkinkan personalisasi layanan, di mana sistem dapat menyesuaikan penyediaan informasi dan dukungan sesuai kebutuhan individu mahasiswa dan seluruh individu akademik, secara signifikan meningkatkan efektivitas dan kualitas dukungan administratif. Dalam konteks layanan, AI memfasilitasi pengembangan berbagai platform teknologi seperti asisten virtual dan sistem interaktif yang mampu memperkaya pengalaman mahasiswa dengan menjadikannya lebih efisien dan responsif. AI memiliki potensi besar untuk meningkatkan kualitas proses pelayanan mahasiswa, menciptakan ekosistem layanan yang lebih efektif, memperluas akses 24 jam terhadap informasi dan sumber daya akademik, serta mempersiapkan mahasiswa untuk berinteraksi dengan infrastruktur digital modern. Pemanfaatan AI juga mendukung otomatisasi proses administratif seperti pendaftaran, pemrosesan dokumen, dan penilaian awal persyaratan, yang mengurangi beban kerja staf administrasi dan memungkinkan penyediaan layanan yang lebih cepat dan akurat, sehingga tercipta lingkungan layanan akademik yang lebih responsif dan berkelanjutan.

Dampak AI terhadap efektivitas layanan akademik telah terbukti memberikan kontribusi signifikan dalam memajukan operasional perguruan tinggi, khususnya pada manajemen data mahasiswa dan peningkatan efisiensi sistem administrasi. Maola (2024) dalam penelitiannya menjelaskan bahwa penerapan AI dalam dunia pendidikan sangat berpengaruh khususnya dalam hal pembelajaran dan sangat memudahkan guru dan siswa dalam menyesuaikan diri dengan kegiatan belajar mengajar, yang menunjukkan bagaimana teknologi ini meningkatkan

adaptabilitas dalam proses pendidikan. Sejalan dengan peningkatan adaptabilitas di ranah pembelajaran tersebut, AI juga meningkatkan kualitas layanan akademik secara keseluruhan. Selain itu, AI memfasilitasi pengelolaan sistem layanan dan dokumentasi digital yang lebih berkualitas dan terjangkau, memperluas akses dan kecepatan layanan akademik. AI juga berperan dalam mengoptimalkan sistem informasi yang membantu dan memungkinkan personalisasi jalur layanan serta memberikan solusi yang efisien, guna meningkatkan kepuasan mahasiswa dan kualitas operasional perguruan tinggi.

2.2.2 Large Language Model (LLM)

Large Language Models (LLM) adalah kategori model *Artificial Intelligence* (AI) yang dicirikan oleh parameter dalam jumlah masif, sering kali mencapai miliaran, dan dilatih pada korpus data teks yang sangat besar. Karakteristik inti LLM adalah kemampuannya untuk memahami, menghasilkan, dan memproses bahasa alami dengan tingkat koherensi dan konteks yang tinggi, menyerupai kognisi manusia dalam tugas-tugas berbahasa. Skala ini memungkinkan aplikasi luas, dari penerjemahan hingga pembuatan konten kreatif. Berdasarkan penelitian oleh OpenAI (2023), peningkatan skala parameter dan data latih secara langsung berkorelasi dengan munculnya zero-shot dan few-shot learning pada model-model tersebut.

Fondasi arsitektur LLM terletak pada Arsitektur *Transformer*, yang diperkenalkan pada tahun 2017. Arsitektur ini menggantikan *Recurrent Neural Networks* (RNN) dan menjauhi *Convolutional Neural Networks* (CNN) dengan mengatasi keterbatasan dalam menangani dependensi jarak jauh secara efisien. Perubahan mendasar ini berpusat pada mekanisme *Self-Attention* yang memungkinkan model secara efisien menimbang pentingnya kata-kata dalam urutan input, terlepas dari jarak fisik di antara kata-kata tersebut. *Kemampuan Self-Attention* ini memungkinkan pemrosesan *input* secara paralel, mempercepat pelatihan model secara dramatis dan menjadikan *transformer* sebagai standar arsitektur untuk hampir semua LLM terkemuka (Vaswani et al., 2017).

Secara umum, proses kerja LLM adalah rangkaian tahapan terintegrasi yang dimulai dengan *tokenization*, di mana teks dipecah menjadi unit diskrit. *Token* diubah menjadi vektor numerik dalam tahap *encoding* sebelum diproses melalui

lapisan-lapisan *attention transformer*. Selama fase pemrosesan, model menggunakan mekanisme *self-attention* yang kompleks untuk menyempurnakan representasi kontekstual vektor. Model modern menerapkan arsitektur modular untuk memperjelas mekanisme kerja internal ini (Wang et al., 2025) guna meningkatkan keandalan dan mengurangi halusinasi. Pada tahap akhir, *Generation*, melibatkan prediksi *token* berikutnya secara *probabilistik* untuk membentuk respons yang koheren. Kapabilitas *Natural Language Processing* (NLP) ini mencakup ringkasan teks, klasifikasi sentimen, menjawab pertanyaan kontekstual, dan menghasilkan narasi atau kode, menjadikan LLM adaptif terhadap berbagai tugas NLP dan memunculkan tantangan sekaligus peluang signifikan dalam bidang AI.

2.2.3 Google Gemini 2.5 Flash

Google Gemini 2.5 Flash adalah salah satu LLM dengan keunggulan pada latensi rendah, dirancang untuk kecepatan superior dan efisiensi biaya yang ideal untuk aplikasi *real-time*. Meskipun memiliki *footprint* yang jauh lebih kecil daripada model unggulan seperti Gemini Pro, Flash mempertahankan kapabilitas *multimodal* yang tangguh, mampu memproses teks, gambar, dan video. Arsitektur yang dioptimalkan, dicapai melalui teknik *quantization* dan *pruning*, memungkinkannya mempertahankan jendela konteks yang sangat besar untuk menganalisis dokumen atau percakapan panjang tanpa mengorbankan kualitas keluaran secara drastis. Kecepatannya menjadikannya pilihan utama untuk asisten virtual interaktif, *chatbot* layanan pelanggan bervolume tinggi, dan peringkasan dokumen saat bepergian, yang membutuhkan biaya inferensi rendah dan waktu respons minimal.

Dari perspektif enterprise, Gemini 2.5 Flash menawarkan keunggulan kompetitif signifikan melalui kemudahan penyebaran via *Application Programming Interfaces* (API), memfasilitasi integrasi langsung ke dalam alur kerja bisnis dan sistem manajemen data internal. Model yang efisien ini sangat mendukung implementasi *Retrieval-Augmented Generation* (RAG), mekanisme yang memungkinkan LLM mendasarkan keluarannya pada data perusahaan yang terverifikasi dan bersifat privat, sehingga menjamin akurasi jawabannya. Pemanfaatan LLM yang dirancang untuk kecepatan, seperti Flash, telah terbukti

menjadi pengubah permainan dalam otomatisasi tugas dan peningkatan efisiensi operasional. (Google, 2025)

2.3 Generative AI dan Natural Language Processing

2.3.1 Generative AI

Generative AI merujuk pada model kecerdasan buatan yang memiliki kemampuan untuk menghasilkan data baru, seperti teks, kode, atau gambar, alih-alih hanya mengklasifikasikannya. Konsep dan prinsip kerja utamanya melibatkan pembelajaran distribusi statistik dari data pelatihan yang sangat besar, memungkinkan model untuk menciptakan sampel data yang otentik dan belum pernah ada sebelumnya. Dalam konteks bahasa, aplikasi *Generative AI* telah secara masif mentransformasi pembuatan konten tekstual, di mana model LLM digunakan untuk membuat artikel, ringkasan, atau email dengan kecepatan yang sangat tinggi. Pergeseran ini, dari AI yang bersifat prediktif menjadi AI yang bersifat kreatif, mengubah arsitektur, strategi, dan nilai yang diciptakan oleh platform digital secara fundamental (Wessel et al., 2025).

Selain kemampuan generatif dasarnya, Generative AI menonjol karena kemampuan adaptasi gaya komunikasi yang luar biasa. Adaptasi gaya ini memungkinkan model untuk memproduksi teks dengan persona atau nada suara yang spesifik, baik itu formal, persuasif, naratif, maupun teknis, hanya dengan panduan *prompt* atau instruksi kontekstual. Dalam penelitian terbaru, kemampuan ini dikategorikan sebagai *Controllable Text Generation* (CTG), yang memandu model untuk menghasilkan teks dengan atribut tertentu sambil mempertahankan kualitas dan kelancaran linguistik (Wang et al., 2024). Fleksibilitas ini sangat krusial dalam penerapan komersial, di mana konten harus disesuaikan secara dinamis untuk berbagai platform dan *audiens*.

2.3.2 Natural Language Processing (NLP)

Natural Language Processing (NLP) adalah fondasi teknis yang memungkinkan *Generative AI* bekerja, terutama ketika berhadapan dengan bahasa dengan morfologi kompleks seperti Bahasa Indonesia. Tantangan NLP dalam konteks Indonesia terletak pada sistem imbuhan yang kaya, di mana satu kata dasar dapat memiliki puluhan bentuk yang berbeda. NLP untuk Bahasa Indonesia harus

mengatasi kompleksitas ini untuk mencapai tingkat akurasi yang memadai. Pengembangan alat dan *resource* khusus, seperti *stemmer* dan *tagger* yang disesuaikan dengan linguistik Indonesia, tetap menjadi fokus utama dalam penelitian di wilayah ini.

Untuk mengatasi kesulitan tersebut, teknik *tokenization* untuk Bahasa Indonesia memerlukan pendekatan yang lebih canggih daripada sekadar membagi kata berdasarkan spasi. Dalam beberapa penelitian, *tokenization* telah berevolusi menjadi pemisahan *multi-word tokens* yang mengakui unit-unit frasa yang memiliki makna tunggal meskipun terdiri dari beberapa kata, daripada hanya memisahkannya menjadi kata tunggal (Petrus et al., 2023). Proses ini sangat penting untuk meningkatkan pemahaman konteks dan semantik model, yang mana model perlu memahami hubungan dan maksud di balik *token*. Pemahaman semantik yang mendalam ini adalah inti dari NLP modern, memastikan bahwa *Generative AI* tidak hanya menghasilkan teks yang lancar secara sintaksis, tetapi juga akurat dan relevan secara makna.

2.4 Teknologi Speech Processing

2.4.1 Speech-to-Text (STT)

Sistem *Speech-to-Text* (STT), atau yang dikenal juga sebagai *Automatic Speech Recognition* (ASR), berfungsi sebagai konverter teknologi yang mengubah sinyal suara menjadi teks tertulis. Proses kerjanya terstruktur dalam tahapan berurutan, dimulai dengan *Audio Preprocessing* untuk penyesuaian sinyal, penghilangan *noise*, dan pemisahan segmen ucapan. Tahap ini diikuti oleh *Feature Extraction*, di mana karakteristik akustik esensial seperti *Mel-Frequency Cepstral Coefficients* (MFCC), *Linear Predictive Coding* (LPC), atau *Perceptual Linear Prediction* (PLP) diekstrak ke dalam representasi vektor numerik (Trivedi et al., 2018).

Inti dari pengenalan ucapan melibatkan *Acoustic Modeling* dan *Language Modeling*. *Acoustic Modeling* menggunakan model statistik seperti *Hidden Markov Models* (HMM), *Deep Neural Networks* (DNN), atau *Deep Belief Networks* (DBN) untuk memetakan vektor fitur akustik ke urutan fonem atau silabel bahasa target (Galatang et al., 2020). Kemudian, *Language Modeling* memprediksi urutan kata yang paling masuk akal berdasarkan konteks linguistik. Dalam konteks aplikasi

asisten virtual di Universitas Ma Chung, teknologi STT menjadi antarmuka *input* pertama kedalam sistem. Ini memungkinkan mahasiswa dan staf berinteraksi menggunakan perintah suara alami untuk mengakses informasi administratif, yang secara signifikan meningkatkan aksesibilitas dan efisiensi operasional.

Meskipun canggih, pengembangan STT untuk Bahasa Indonesia menghadapi tantangan spesifik, terutama karena variasi dialek regional dan fenomena pencampuran Bahasa Indonesia dengan Bahasa Inggris atau bahasa daerah yang umum terjadi, bahkan di lingkungan kampus. Kinerja sistem ini akan dievaluasi menggunakan metrik standar industri *Word Error Rate* (WER), yang mengukur persentase kesalahan kata termasuk *substitution errors*, *deletion errors*, dan *insertion errors*. Pengujian ini penting dilakukan pada berbagai kondisi nyata, untuk memastikan suara yang akan menjadi *input* tidak melenceng dari pengucapan aslinya.

2.4.2 Text-to-Speech (TTS)

Text-to-Speech (TTS) adalah teknologi vital yang berfungsi mengonversi teks tertulis menjadi *output* audio berupa suara sintetis menggunakan *engine TTS* (Kurniadi et al., 2022). Sistem TTS memungkinkan komputer berkomunikasi dengan pengguna secara intuitif melalui suara dan bekerja melalui serangkaian komponen terintegrasi. Proses ini dimulai dengan *Text Analysis*, yang melakukan *parsing* dan normalisasi teks yang akan diucapkan. Selanjutnya, *Linguistic Processing* menganalisis struktur bahasa menggunakan teknik seperti *part-of-speech tagging* dan *shallow parsing* untuk menentukan batas frasa dan struktur sintaksis yang sangat memengaruhi intonasi ucapan yang akan dihasilkan.

Setelah pemrosesan linguistik, komponen *Acoustic Modeling* memprediksi dan menghasilkan parameter akustik yang merepresentasikan karakteristik suara, berdasarkan fitur-fitur linguistik yang telah diekstrak. Kemudian, *Audio Synthesis* adalah tahap yang mengonversi parameter akustik ini menjadi *waveform* audio final yang dapat didengar, dengan mempertimbangkan aspek intonasi, durasi, dan intensitas untuk menghasilkan suara yang natural. Proses sintesis ini sering menggunakan berbagai metode seperti *concatenative synthesis*, *formant synthesis*, atau *articulatory synthesis* untuk menciptakan representasi suara akhir. Dalam implementasinya, khususnya pada asisten virtual untuk layanan akademik

Universitas Ma Chung, teknologi TTS berfungsi sebagai *interface output* untuk memberikan respons audio, memungkinkan interaksi *hands-free* yang lebih natural dan *user-friendly* dibandingkan teks semata.

Pada TTS, *naturalness* adalah aspek yang sangat penting karena menentukan seberapa mirip suara sintetis dengan ucapan manusia, serta seberapa mudah ia dipahami. Kualitas *naturalness* dipengaruhi oleh beberapa faktor utama, yaitu: intonasi, ritma, pola penekanan kata dan kelancaran dalam transisi antar *phoneme* atau *syllable*. Implementasi TTS modern bahkan dapat mengoptimalkan suara yang dikeluarkan dengan menggunakan berbagai dialek bahasa melalui mekanisme *language identifier* yang meningkatkan akurasi dan kualitas *output*. Evaluasi kualitas suara dilakukan melalui pengujian subjektif di mana pengguna menilai aspek kejelasan (*intelligibility*), kealamian (*naturalness*), dan kelancaran (*smoothness*), serta melalui pengukuran akurasi sistem dalam mengonversi berbagai jenis teks. Oleh karena itu sistem ini akan diuji menggunakan metode *blackbox test* dengan matriks *Mean Opinion Score* (MOS).

2.4.3 F5-TTS

F5-TTS (*Fairytaler that Fakes Fluent and Faithful Speech*) adalah sistem *text-to-speech* (TTS) yang diperkenalkan oleh Chen et al. (2024), beroperasi sepenuhnya secara *non-autoregresif* menggunakan teknologi *flow matching* dengan arsitektur *Diffusion Transformer* (DiT). Keunggulan arsitekturnya terletak pada penyederhanaan radikal. sistem ini mengeliminasi kebutuhan akan komponen kompleks seperti *duration model*, *text encoder* khusus, dan *phoneme alignment* yang lazim pada TTS tradisional. *Input* teks hanya diberi *padding* dengan *filler tokens* dan diprediksi secara langsung. F5-TTS menggunakan *ConvNeXt v2* sebagai *backbone* dan menerapkan *adaptive root mean square normalization* untuk stabilitas pelatihan. Melalui *flow matching*, model secara efisien memetakan distribusi *noise gaussian* ke distribusi data suara, menciptakan pendekatan *end-to-end* yang jauh lebih ringkas. Sifat *non-autoregresif* ini juga menghasilkan efisiensi inferensi yang tinggi karena memungkinkan generasi audio secara paralel.

Kemampuan *zero-shot voice cloning* adalah fitur utama F5-TTS yang membedakannya, memungkinkan sistem untuk mereplikasi suara pembicara baru

hanya dengan beberapa detik *reference* audio tanpa memerlukan data pelatihan tambahan. Sistem ini dapat menduplikasi suara berkualitas tinggi dengan mempertahankan intonasi dan timbre dengan mengintegrasikan representasi *speaker embedding* ke dalam proses *flow matching*. Dibandingkan dengan sistem TTS sebelumnya, F5-TTS menawarkan kualitas audio setara *state-of-the-art* dengan kompleksitas sistem yang jauh lebih rendah. Tetapi adaptasi F5-TTS untuk bahasa lokal, seperti Bahasa menghadapi tantangan terutama pada penanganan perbedaan fonetik. F5-TTS saat ini cenderung lebih menguasai duplikasi suara dalam bahasa Inggris dan bahasa Mandarin. Hasilnya jika mengucapkan bahasa lain terutama bahasa Indonesia, hasil fonetiknya akan terpengaruh dengan fonetik dari bahasa Inggris.

2.4.4 Chatterbox

Chatterbox TTS merupakan teknologi text-to-speech berbasis kecerdasan buatan yang dirancang untuk menghasilkan suara sintetis dengan kualitas yang natural dan mudah dipahami. Sistem ini tidak hanya mengonversi teks menjadi suara, tetapi juga menganalisis struktur kalimat, konteks, serta emosi yang tersirat di dalam teks untuk menentukan intonasi, jeda, dan penekanan kata yang tepat. Dengan pendekatan ini, suara yang dihasilkan terdengar lebih hidup dan tidak monoton, sehingga cocok digunakan pada aplikasi yang menuntut interaksi jangka panjang dengan pengguna. Selain itu, Chatterbox TTS dikembangkan agar mampu menangani teks yang panjang dan kompleks tanpa kehilangan konsistensi kualitas suara. Hal ini membuatnya relevan untuk kebutuhan narasi, pembelajaran daring, maupun sistem informasi berbasis suara.

Salah satu keunggulan utama Chatterbox TTS terletak pada model multilingual yang dimilikinya, di mana satu arsitektur model dapat mendukung berbagai bahasa secara bersamaan. Model ini dilatih menggunakan data multibahasa sehingga mampu memahami perbedaan fonetik, tata bahasa, dan pola intonasi dari setiap bahasa yang didukung. Dengan pendekatan multilingual, Chatterbox TTS dapat melakukan transisi antarbahasa dengan lebih mulus serta mengurangi kebutuhan pembuatan model terpisah untuk setiap bahasa. Keunggulan ini sangat bermanfaat bagi aplikasi global yang melayani pengguna dari berbagai latar belakang bahasa, karena efisiensi pengembangan dan konsistensi kualitas

suara dapat tetap terjaga. Selain itu, model multilingual juga membuka peluang untuk pengembangan fitur lanjutan seperti code-switching, yaitu kemampuan mengucapkan lebih dari satu bahasa dalam satu kalimat secara alami.

2.5 Retrieval-Augmented Generation (RAG)

2.5.1 Konsep dan Arsitektur RAG

Retrieval-Augmented Generation (RAG) adalah sebuah teknik canggih untuk membuat model generatif seperti chatbot mampu menghasilkan jawaban yang lebih akurat dan tervalidasi. Cara kerja RAG dimulai saat menerima pertanyaan (*query*) dari pengguna, kemudian sistem akan mencari dokumen yang relevan dari *knowledge base* menggunakan teknik *similarity search*. Terakhir, sistem akan menggabungkan konteks (*augmented*) dari dokumen yang ditemukan ini dengan *Large Language Model* (LLM) untuk membuat jawaban yang sesuai konteks, dan faktual (Samudra et al., 2025). Ini berbeda dari LLM murni, yang hanya mengandalkan pengetahuan dari proses pelatihan awal, karena RAG mengurangi risiko *hallucination* atau pemberian informasi yang tidak akurat dengan mengakses data eksternal secara dinamis.

RAG membutuhkan tempat penyimpanan berupa *vector database* untuk memrepresentasi numerik (*embedding*) dari semua dokumen, memungkinkan pencarian semantik yang efisien. Proses *Similarity Search* kemudian menemukan dokumen yang paling mirip dengan *query* pengguna, sering kali menggunakan metrik seperti *cosine similarity* atau *euclidean distance* pada ruang vektor, di mana penggunaan indeks *Hierarchical Navigable Small World* (HNSW) terbukti meningkatkan efisiensi pencarian pada dataset besar. Tahap *Context Augmentation* adalah saat dokumen yang relevan digabungkan dengan *query* menjadi *augmented prompt* yang kaya konteks.

Tahap terakhir adalah *Response Generation*, di mana LLM memproses *augmented prompt* untuk menghasilkan jawaban yang natural dan akurat berdasarkan konteks yang disajikan. Kualitas jawaban dievaluasi menggunakan metrik seperti Faithfulness (seberapa setia jawaban terhadap dokumen sumber) dan Answer Relevancy (relevansi jawaban terhadap pertanyaan pengguna). Keunggulan utama RAG terletak pada kemampuannya untuk mengurangi fenomena *hallucination* atau model bias pada aplikasi domain-spesifik, seperti

layanan akademik di Universitas Ma Chung. Dengan adanya grounding pada dokumen faktual (misalnya panduan akademik dan prosedur kampus), RAG memastikan respons yang diberikan kepada mahasiswa dan staf selalu akurat, terverifikasi, dan up-to-date.

2.5.2 Vector Database dan Embedding

Vektor Database adalah sistem manajemen basis data khusus yang dirancang untuk menyimpan, mengindeks, dan mencari representasi vektor berdimensi tinggi dari data yang tidak terstruktur seperti teks atau gambar. Berbeda dari basis data tradisional yang hanya mencari kecocokan kata, *Vektor Database* memungkinkan pencarian kemiripan semantik (*similarity search*) dengan menghitung skor jarak antar vektor dalam ruang multidimensi. Proses ini sangat penting untuk *Retrieval-Augmented Generation* (RAG) karena memungkinkan sistem menemukan dokumen yang paling relevan berdasarkan makna pertanyaan, bukan sekadar kata kunci, dengan bantuan algoritma *Approximate Nearest Neighbor* (ANN) seperti *Hierarchical Navigable Small World* (HNSW) untuk efisiensi tinggi. Dalam konteks asisten virtual akademik, basis data ini menyimpan *embeddings* dalam vektor numerik dari semua dokumen akademik, memungkinkan respons yang cepat dan akurat.

Supabase, sebuah platform *open-source Backend-as-a-Service*, menyediakan solusi praktis untuk mengimplementasikan *vektor database* melalui ekstensi *pgvector* pada basis data *PostgreSQL*, yang mendukung pencarian kemiripan vektor secara native. Keunggulan penggunaan Supabase dalam proyek asisten virtual akademik adalah kemudahan integrasi melalui RESTful API dan fitur *authentication*, serta kemampuan *real-time* untuk segera memperbarui *knowledge base* saat ada perubahan kebijakan. Proses Embedding adalah inti dari pencarian semantik, yaitu transformasi data tekstual menjadi vektor numerik padat (*dense vector*) berdimensi tinggi seperti 768 atau 1024 dimensi yang secara akurat menangkap makna dan informasi kontekstual dari teks tersebut. Pencarian kemiripan dilakukan dengan menghitung skor kemiripan (*similarity score*) antara vektor pertanyaan (*query*) dan vektor dokumen (*document embeddings*), seringkali menggunakan metrik *cosine similarity* (mengukur sudut antar vektor), di mana dokumen dengan skor kemiripan tertinggi dianggap paling relevan.

Optimasi kinerja *retrieval* dalam RAG adalah kunci untuk meningkatkan akurasi dan kecepatan. Salah satu caranya adalah mengoptimalkan ukuran *chunk* panjang dokumen menjadi bagian-bagian yang ukurannya optimal. Hal ini dikarenakan *chunk* yang terlalu besar dapat mengurangi presisi karena mencakup informasi tak relevan, sementara terlalu kecil dapat menghilangkan konteks penting. Untuk mempermudah proses *chunk*, dokumen perlu disusun secara terstruktur sehingga dapat terlihat poin inti informasi dengan detail informasinya. Selain itu untuk meningkatkan keakuratan perlu dilakukan *reranker* yang bertujuan untuk memberikan *ranking* ulang untuk setiap hasil *similarity search* yang diperoleh untuk dipilih hasil yang memiliki *ranking* paling tinggi.

2.5.3 Integration Framework

n8n adalah platform *workflow automation open-source* yang sangat berperan penting, memungkinkan integrasi berbagai aplikasi, servis, dan API melalui antarmuka visual berbasis *node*. Platform ini memfasilitasi *orchestration* kompleks dari *multiple services* tanpa memerlukan *extensive coding*. Dalam konteks pengembangan asisten virtual akademik, n8n berfungsi sebagai lapisan integrasi yang menghubungkan komponen-komponen utama sistem, seperti *chatbot interface*, *Large Language Model API*, *vector database*, *traditional relational database* untuk data mahasiswa, dan external APIs dari sistem informasi akademik yang sudah ada. Keunggulan n8n terletak pada kemampuannya membuat workflow kompleks dengan *branching logic*, *conditional execution*, *error handling*, dan *data transformation* melalui antarmuka *drag-and-drop*, yang secara signifikan mempercepat *development* dan mempermudah *maintenance* dibandingkan skrip integrasi yang dikodekan khusus. Platform ini mendukung ratusan integrasi *pre-built* dan menyediakan *HTTP Request nodes* serta *webhook triggers* yang memungkinkan integrasi dengan *custom APIs* atau *legacy systems* yang mungkin ada di lingkungan universitas.

Untuk memastikan n8n *workflow engine* dapat berjalan secara konsisten, *portable*, dan terisolasi di berbagai lingkungan, platform ini idealnya di-deploy menggunakan Docker. Docker adalah platform *containerization* yang memungkinkan *packaging* aplikasi beserta semua *dependencies*, *libraries*, dan *configurations* ke dalam container yang *portable*, *lightweight*, dan *isolated*,

sehingga aplikasi dapat berjalan secara konsisten dari *development* hingga *production*. Teknologi *containerization* Docker membuat proses pemaketan aplikasi bersama komponennya secara cepat dalam sebuah container yang terisolasi, sehingga dapat dijalankan dalam infrastruktur lokal tanpa melakukan perubahan konfigurasi pada container, dan Docker juga sangat ringan dan cepat jika dibandingkan dengan mesin virtual yang berbasis *hypervisor* (Adiputra, 2016).

Dalam arsitektur *multi-service* asisten virtual akademik, n8n menjalankan fungsinya sebagai *integration* layer dengan memanfaatkan berbagai pola integrasi API. RESTful API menjadi pola dominan untuk komunikasi sinkron antar-layanan melalui HTTP-based endpoints dengan JSON sebagai format data (Faqihuddin et al., 2025). Secara spesifik, *workflow* n8n sering kali dipicu oleh *webhook*, yang merupakan bagian dari pola *event-driven architecture*. Di sini, *webhook trigger* n8n berfungsi sebagai titik masuk *event* yang memungkinkan *services* berkomunikasi secara asinkron ketika ada kejadian tertentu, seperti permintaan baru dari *chatbot interface* yang diteruskan ke n8n melalui API integration. Pola ini memungkinkan *loose coupling* dan skalabilitas yang lebih baik, terutama untuk operasi yang memakan waktu seperti *document embedding* atau *batch data processing*. Selain itu, API *Gateway pattern* juga penting, di mana sebuah *gateway service* bertindak sebagai *single entry point* yang menangani *routing* dan *authentication* sebelum meneruskan permintaan, termasuk *webhook* n8n, ke *backend services* yang sesuai. Implementasi integrasi yang baik ini memerlukan perhatian terhadap error handling, *retry mechanisms*, *circuit breakers* untuk *fault tolerance*, serta *comprehensive logging* dan *monitoring*, di mana penerapan praktik DevOps dengan *continuous monitoring* dapat meningkatkan stabilitas dan *reliability* sistem API secara signifikan.

2.6 Sistem Informasi Akademik

2.6.1 Karakteristik Layanan Akademik Universitas

Layanan akademik universitas mencakup berbagai jenis informasi yang dibutuhkan mahasiswa untuk menunjang kelancaran proses pendidikan di perguruan tinggi. Sistem informasi akademik menyediakan kemudahan komunikasi dan penyampaian informasi antara universitas dan mahasiswa yang mencakup data akademik, kurikulum, jadwal kuliah, informasi ujian, modul studi, progres

akademik, dan pendaftaran mata kuliah. Informasi akademik juga meliputi aspek administratif seperti prosedur pendaftaran, persyaratan dokumen, informasi biaya pendidikan, dan mekanisme pengajuan layanan administratif lainnya. Menurut Kambau (2024), transformasi digital pada perguruan tinggi di Indonesia mencakup penerapan teknologi dalam penelitian, administrasi, dan layanan mahasiswa dengan tujuan menciptakan lingkungan pendidikan yang lebih fleksibel, inklusif, dan responsif terhadap kebutuhan *stakeholders*.

Mahasiswa memiliki kebutuhan mendasar untuk mengakses informasi akademik secara cepat, mudah, dan akurat dalam mendukung proses pembelajaran mereka di perguruan tinggi. Kebutuhan akses informasi meliputi kemampuan untuk memeriksa status akademik, mengunduh materi pembelajaran, mendaftar mata kuliah, melihat jadwal perkuliahan dan ujian, serta mengakses berbagai layanan administratif. Mahasiswa mengharapkan sistem informasi akademik yang dapat diakses kapan saja dan dari mana saja melalui berbagai perangkat dengan *interface* yang *user-friendly*, navigasi yang intuitif, dan respons sistem yang cepat.

Layanan akademik tradisional menghadapi berbagai tantangan dalam memenuhi ekspektasi mahasiswa yang semakin tinggi di era digital saat ini. Layanan konvensional yang mengandalkan interaksi tatap muka dan dokumen fisik memerlukan sumber daya yang cukup besar untuk menangani banyaknya permintaan ketersediaan informasi dari mahasiswa. Keterbatasan jam operasional layanan, antrian panjang, dan proses manual yang memakan waktu menjadi hambatan utama dalam efisiensi layanan akademik. Sistem tradisional juga rentan terhadap kesalahan manusia dalam pengelolaan data, kurang mampu menyediakan informasi *real-time*, dan mengalami kesulitan dalam mengintegrasikan berbagai jenis layanan administratif dan akademik dalam satu sistem yang koheren.

2.6.2 Digitalisasi Layanan Akademik

Transformasi digital dalam pendidikan tinggi merupakan perubahan fundamental yang mengintegrasikan teknologi informasi ke dalam seluruh aspek layanan akademik dan operasional institusi. Menurut penelitian Widiastuti et al (2025), transformasi digital di sektor pendidikan tinggi mendorong perlunya sistem manajemen yang terintegrasi guna mendukung peningkatan mutu layanan kepada seluruh *stakeholder*. Proses transformasi ini mencakup digitalisasi pembelajaran,

penelitian, administrasi, dan layanan mahasiswa dengan tujuan menciptakan ekosistem pendidikan yang lebih adaptif dan inovatif. mengidentifikasi bahwa faktor-faktor seperti kualitas teknologi yang digunakan, komitmen pimpinan, kesiapan sumber daya manusia, dukungan kebijakan eksternal, serta ketersediaan infrastruktur menjadi determinan utama keberhasilan implementasi transformasi digital di perguruan tinggi.

Sistem informasi akademik terintegrasi merupakan platform komprehensif yang menghubungkan berbagai fungsi dan layanan akademik dalam satu ekosistem digital yang terkoordinasi. Menurut penelitian Widiastuti et al (2025), Sistem Informasi Manajemen Mutu Terpadu berkontribusi signifikan dalam efisiensi administrasi akademik, transparansi data, peningkatan akuntabilitas, serta pengambilan keputusan berbasis bukti. Integrasi sistem memungkinkan aliran informasi yang lancar antar berbagai unit layanan seperti registrasi, akademik, keuangan, perpustakaan, dan kemahasiswaan.

User experience dalam layanan digital akademik menjadi faktor krusial yang menentukan tingkat adopsi dan kepuasan pengguna terhadap sistem informasi yang dikembangkan. Parameter pengukuran *user experience* mencakup aspek kegunaan sistem (*usefulness*), kemudahan penggunaan (*ease of use*), kemudahan pembelajaran (*ease of learning*), dan kepuasan pengguna (*satisfaction*). Hasil penelitian. Evaluasi *user experience* secara berkala dengan melibatkan mahasiswa sebagai responden menjadi praktik penting untuk memastikan sistem terus relevan dan memenuhi ekspektasi pengguna.

2.7 Arsitektur dan Integrasi Sistem

2.7.1 Arsitektur Microservices

Arsitektur *microservices* merupakan pendekatan pengembangan perangkat lunak yang membagi aplikasi menjadi layanan-layanan kecil dan independen yang dapat dikembangkan, diimplementasikan, dan diskalakan secara terpisah. Prinsip utama dalam desain *microservices* mencakup pemisahan fungsi berdasarkan domain bisnis, enkapsulasi data untuk setiap *service*, dan komunikasi antar *service* melalui API yang terdefinisi dengan baik. Implementasi *microservices* dalam sistem asisten virtual memungkinkan pembagian fungsi utama seperti *webapp*,

dengan *service* untuk asisten virtual LLM dan TTS yang terpisah sehingga meningkatkan fleksibilitas efisiensi, dan kemudahan pemeliharaan sistem.

API *Gateway* merupakan komponen krusial dalam arsitektur *microservices* yang berfungsi sebagai pintu masuk tunggal untuk semua permintaan klien ke berbagai *microservices* di *backend*. Menurut Novianto dan Munir (2022), dengan semakin besar dan kompleksnya suatu sistem informasi, kebutuhan akan pengolahan dan integrasi data menjadi perhatian besar bagi banyak organisasi termasuk institusi pendidikan tinggi. API *Gateway* mengelola *routing request* ke *service* yang tepat, melakukan agregasi *response* dari *multiple services*, dan mengimplementasikan cross-cutting concerns seperti autentikasi, authorization, *rate limiting*, dan *logging*. Service *orchestration* mengatur koordinasi dan urutan eksekusi antar berbagai *microservices* untuk menyelesaikan suatu *business process* yang kompleks, seperti proses registrasi mahasiswa yang melibatkan validasi data, pengecekan kuota, pembayaran, dan pembuatan akun secara terkoordinasi.

Scalability merupakan salah satu keuntungan utama dari arsitektur *microservices* yang memungkinkan sistem untuk menangani peningkatan beban kerja dengan cara yang efisien dan *cost-effective*. Dalam konteks sistem informasi akademik, *scalability* memungkinkan institusi untuk menambah kapasitas layanan tertentu yang mengalami *high traffic* seperti *service* pendaftaran pada periode registrasi tanpa perlu mengubah seluruh aplikasi. Menurut Irawan dan Fauzi (2025), sistem yang dibangun dengan arsitektur *microservices* terbukti mudah dipelihara dengan membagi fungsi utama sistem ke dalam layanan independen yang memungkinkan *bug fixes* dan *feature updates* dilakukan pada individual *service* tanpa mengganggu operasional *service* lainnya. *Maintainability* ditingkatkan melalui *modularitas* dan isolasi antar *service* yang memudahkan proses *development*, *testing*, dan *deployment* dengan setiap *service* dapat menggunakan *technology stack* yang paling sesuai dengan kebutuhannya.

2.7.2 Web Application Framework

PHP MVC (*Model-View-Controller*) merupakan *patern* arsitektur yang memisahkan aplikasi menjadi tiga komponen utama untuk meningkatkan organisasi kode dan memfasilitasi pengembangan yang terstruktur. Menurut Rahmawati dan Sumarsono (2024), salah satu web *framework* paling populer adalah Laravel yang

berbasis bahasa pemrograman PHP dengan dukungan arsitektur *Model View Controller* (MVC) yang memisahkan layer model, *view*, dan *controller* sehingga membantu *developers* memiliki pemahaman yang jelas tentang semua modul dalam aplikasi. Model bertanggung jawab untuk pengelolaan data dan *business logic*, *View* menangani *presentation layer* dan *user interface*, sedangkan *Controller* berfungsi sebagai penghubung yang memproses *request* dan mengkoordinasikan *Model* dan *View*. Implementasi MVC dalam framework PHP memberikan struktur yang standar dan best practices untuk pengembangan aplikasi web yang *maintainable* dan scalable dengan *features* seperti *routing*, *middleware*, dan *ORM* yang mempercepat *development cycle*.

Pemisahan *frontend* dan *backend* merupakan praktik modern dalam pengembangan web *application* yang memungkinkan *development* yang independen dan fleksibilitas dalam memilih teknologi untuk setiap layer. Backend berfokus pada *business logic*, *data processing*, dan *integration* dengan sistem eksternal, sementara *frontend* menangani *user interface* dan *user experience* dengan komunikasi antara keduanya dilakukan melalui API yang terdefinisi dengan baik menggunakan format JSON. Penggunaan *framework* dalam pengembangan aplikasi *website* dapat mempercepat waktu pengembangan dengan mengurangi jumlah kode yang harus ditulis serta mempermudah integrasi dengan berbagai layanan pihak ketiga. Pendekatan ini memungkinkan satu *backend* untuk melayani *multiple frontend platforms* seperti *web application*, dan *voice interfaces* untuk asisten virtual, dengan *separation of concerns* meningkatkan *scalability* dan memfasilitasi *parallel development* oleh tim yang berbeda.

Real-time communication protocols memungkinkan pertukaran data yang *instant* antara *client* dan server untuk mendukung aplikasi yang memerlukan *update* data secara *real-time* tanpa *polling*. *WebSocket* merupakan *protocol* yang paling umum digunakan untuk komunikasi *bidirectional full-duplex* antara browser dan server, memungkinkan server untuk *push* data ke *client* tanpa *client* harus *request* terlebih dahulu, yang sangat penting untuk asisten virtual dalam memberikan *responses* yang *instant* dan natural. *Server-Sent Events* (SSE) merupakan alternatif untuk *one-way communication* dari server ke *client* yang lebih *simple* namun memadai untuk banyak *use cases* seperti *streaming responses* dari LLM. Protocol

seperti *WebSocket* memungkinkan *Maintain persistent connection* yang efisien untuk *continuous conversation* dengan asisten virtual, dengan *proper implementation* termasuk handling connection failures, reconnection logic, dan *message queuing* untuk *ensure reliable delivery* dari *responses* kepada *users*.

2.7.3 Integration Patterns

RESTful API merupakan *architectural style* untuk *designing networked applications* yang menggunakan HTTP *methods* untuk melakukan operasi CRUD (*Create, Read, Update, Delete*) pada *resources*. Menurut Novianto dan Munir (2022), RESTful API menawarkan kemudahan dalam menjembatani pertukaran data tanpa mempermasalahkan perbedaan platform dan bahasa pemrograman yang digunakan dalam *development*. Prinsip REST mencakup *stateless communication*, *uniform interface*, *resource-based URLs*, dan penggunaan standard HTTP *methods* (GET, POST, PUT, DELETE) dengan desain yang baik mengikuti konvensi penamaan yang konsisten, menggunakan HTTP status *codes* yang *appropriate*, dan menyediakan *clear error messages*. Implementasi RESTful API dalam sistem informasi akademik memungkinkan berbagai aplikasi dan services untuk mengakses dan memanipulasi data akademik secara *standardized*, dengan API desain yang baik mencakup versi untuk *backward compatibility*, *pagination* untuk *large datasets*, serta *comprehensive documentation*.

Webhook merupakan mekanisme penting untuk memfasilitasi komunikasi berbasis kejadian dan notifikasi waktu nyata antar sistem, di mana server secara proaktif mengirimkan permintaan HTTP POST ke URL yang terkonfigurasi saat suatu *event* terjadi berbeda dengan *polling* yang harus berulang kali meminta pembaruan. Penerapan *webhook* harus fokus pada keandalan dan keamanan *endpoint webhook* harus siap menangani permintaan konkuren, memvalidasi tanda tangan *payload*, dan merespons cepat untuk menghindari *timeout*. Dalam konteks akademik, *webhook* memungkinkan asisten virtual menerima permintaan seketika dari pengguna, sehingga dapat secara *realtime* memberikan layanan.

Pembangunan integrasi yang andal dan tahan banting terhadap kegagalan sangat bergantung pada penanganan *error* yang komprehensif dan mekanisme coba ulang yang cerdas. Penanganan *error* yang baik melibatkan penggunaan kode status HTTP yang tepat, penyediaan pesan *error* yang mendetail untuk membantu

debugging, dan menerapkan *graceful degradation* (penurunan fungsionalitas yang elegan) untuk mempertahankan fungsionalitas parsial saat beberapa layanan tidak tersedia, bahkan dengan penerapan pola *circuit breaker* untuk mencegah kegagalan beruntun. Sementara itu, mekanisme *retry* dengan strategi seperti *exponential backoff* dan *jitter* sangat penting untuk mengatasi kegagalan sementara, seperti *network timeout*, tanpa membebani sistem. Strategi ini mencakup penentuan operasi yang aman untuk dicoba ulang, penetapan batas maksimum percobaan ulang, dan penggunaan strategi *backoff* yang sesuai. Sebagai contoh, menurut Sinlae et al. (2024), menawarkan dokumentasi dan dukungan komunitas yang kuat untuk mempermudah implementasi penanganan *error*. Dalam konteks asisten virtual akademik, penanganan *error* yang kuat menjamin pengalaman pengguna tetap baik meskipun layanan *backend* bermasalah, di mana asisten mampu menyampaikan pesan yang jelas mengenai tidak tersedia sementara dan secara otomatis mencoba ulang operasi yang diperlukan.

2.8 Evaluasi dan Metrik Kinerja

2.8.1 Metrik Akurasi Sistem

Word Error Rate (WER) merupakan metrik standar yang digunakan untuk mengukur performa sistem pengenalan suara otomatis dengan menghitung persentase kesalahan dalam transkripsi kata yang dihasilkan. WER dihitung berdasarkan jumlah substitusi (kata yang salah dikenali), insersi (kata yang ditambahkan), dan *delesi* (kata yang hilang) dibandingkan dengan transkrip referensi yang dibuat manusia. Metrik ini berkisar dari 0 hingga 1, dimana 0 mengindikasikan transkrip yang identik sempurna dan 1 menunjukkan transkrip yang sama sekali berbeda tanpa kesamaan. Dalam konteks asisten virtual untuk layanan akademik, WER yang rendah sangat penting untuk memastikan sistem dapat memahami pertanyaan dan perintah mahasiswa dengan akurat, terutama untuk istilah-istilah akademik spesifik yang mungkin tidak umum dalam *vocabulary* standar. Untuk melakukan perhitungannya digunakan rumus dengan S atau *substitution* yang menghitung berapa banyak kata yang salah, D atau *delegation* yang menghitung berapa kata yang hilang, I atau *insertions* yang menghitung berapa kata yang ditambahkan dan yang seharusnya dan N adalah total jumlah kata.

$$WER = \frac{S+D+I}{N} \quad (2-1)$$

Mean Opinion Score (MOS) merupakan metrik subjektif yang digunakan untuk mengevaluasi kualitas suara yang dihasilkan oleh sistem *Text-to-Speech* dengan melibatkan penilaian dari sejumlah pendengar manusia. Penilaian MOS menggunakan skala 1 hingga 5, dimana 1 menunjukkan kualitas yang sangat buruk dan 5 menunjukkan kualitas yang sangat baik atau natural seperti suara manusia. Evaluasi mencakup aspek-aspek seperti *naturalness* (kealamian), *intelligibility* (kejelasan), dan *pleasantness* (kenyamanan) dari suara yang dihasilkan. Dalam implementasi asisten virtual akademik, MOS yang tinggi penting untuk memastikan respons audio yang dihasilkan mudah dipahami, sehingga meningkatkan *user experience* secara keseluruhan. Untuk pengukurannya menggunakan rumus dengan N adalah total jumlah responden, R adalah nilai atau *opinion score*.

$$MOS = \frac{\sum_{i=1}^N R_i}{N} \quad (2-2)$$

Likert Scale Evaluation merupakan metode evaluasi yang digunakan untuk menilai kualitas respons asisten virtual berdasarkan persepsi pengguna. Dalam pendekatan ini, pengguna diminta memberikan penilaian terhadap beberapa aspek kinerja sistem menggunakan skala Likert, dengan poin dari 1 hingga 5, di mana nilai rendah menunjukkan sangat buruk dan nilai tinggi menunjukkan sangat baik. Metode ini berfokus pada penilaian subjektif terhadap aspek tingkat keakuratan jawaban, tingkat relevansi jawaban, tingkat kelengkapan jawaban, tingkat kejelasan jawaban dan tingkat kepuasan atas jawaban yang diberikan. Penilaian kemudian diolah secara kuantitatif dengan menghitung total dan rata-rata skor (R) yang diberikan oleh seluruh responden (N). Semakin tinggi skor rata-rata yang diperoleh, semakin baik kualitas respons yang dirasakan oleh pengguna.

$$Likert = \frac{\sum_{i=1}^N R_i}{N} \quad (2-3)$$

2.8.2 User Experience Metrics

Usability testing merupakan metode pengujian yang digunakan untuk mengukur seberapa mudah sistem dapat digunakan dan kendala yang ditemukan saat penggunaannya untuk dapat dievaluasi dan menghasilkan data yang relevan. Menurut penelitian Dyayu et al (2023), *usability* testing memiliki beberapa metrik

yang diukur yaitu *learnability* (kemudahan mempelajari sistem), *efficiency* (efisiensi dalam menyelesaikan tugas), *memorability* (kemudahan mengingat cara penggunaan), *error rate* (tingkat kesalahan pengguna), dan *satisfaction* (kepuasan pengguna). *Framework* pengujian *usability* mencakup perancangan skenario yang mencerminkan aktivitas pengguna dalam keadaan sebenarnya dan melibatkan partisipan yang mewakili target *users*. Dalam konteks asisten virtual akademik, *usability* testing penting untuk memastikan mahasiswa dapat dengan mudah berinteraksi dengan sistem melalui perintah suara untuk mengakses informasi akademik tanpa mengalami kesulitan atau frustasi.

SUS akan diambil skornya dengan memberikan 10 pernyataan dimana setiap pernyataan ganjil berupa pernyataan positif dan pernyataan genap adalah pernyataan negatif. Responden akan memberikan nilai 1 hingga 5 dimana 1 mempresentasikan sangat tidak setuju, 2 tidak setuju, 3 netral, 4 setuju dan 5 sangat setuju. Menurut penelitian Prabowo (2021), *System Usability Scale* (SUS) merupakan alat pengujian yang dikembangkan dengan menerapkan sepuluh pertanyaan yang memberikan pandangan global secara subjektif mengenai ketergunaan sistem. Metode SUS menghasilkan skor dengan rentang 0-100, dimana skor di atas 68 dikategorikan sebagai diatas rata-rata dan skor di atas 80 dikategorikan sebagai *excellent*. *Assessment* ini memberikan wawasan tentang aspek-aspek sistem yang perlu diperbaiki berdasarkan *feedback* pengguna langsung.

Cara melakukan perhitungan SUS dengan menghitung skor setiap pertanyaan dahulu (Q_i), kemudian ditotal untuk setiap respondennya (R_i) dan terakhir baru dilakukan rata-rata dari semua skor responden (SUS) untuk mendapatkan skor akhir. Secara detailnya adalah sebagai berikut.

1. Pernyataan ganjil yang mempresentasikan pernyataan positif akan dihitung dengan skor pernyataan responden dikurangkan dengan 1.

$$Q_i = \text{skor} - 1 \quad (2-4)$$

2. Pernyataan genap yang mempresentasikan pernyataan negatif akan dihitung dengan 5 dikurangkan dengan skor pernyataan responden

$$Q_i = 5 - \text{skor} \quad (2-5)$$

3. Setelah setiap skor responden diperoleh, selanjutnya dilakukan perhitungan skor akhir setiap responden.

$$R_i = \sum_{i=1}^{10} Q_i \quad (2-6)$$

4. Sebelum dirata-rata dilakukan penkalian 2,5 untuk setiap skor responden

$$R_i = R_i \times 2,5 \quad (2-7)$$

5. Setelah mendapatkan skor akhir setiap responden maka dilakukan rata-rata skor akhir

$$SUS = \frac{\sum_{i=1}^n R_i}{n} \quad (2-8)$$

2.9 Blackbox Testing

Blackbox testing adalah metode pengujian sistem aplikasi yang berfokus pada pengujian fungsi sistem tanpa memperhatikan struktur internal atau kode sumber yang ada di dalamnya. Pengujian ini dilakukan dengan cara memberikan *input* tertentu ke dalam sistem dan kemudian mengamati *output* yang dihasilkan. Tujuan utama dari Blackbox testing adalah memastikan bahwa sistem bekerja sesuai dengan kebutuhan dan spesifikasi yang telah ditentukan. Dengan pendekatan ini, penguji tidak perlu memahami logika pemrograman internal, sehingga metode ini dapat dilakukan oleh tester maupun pengguna akhir.

Dalam penerapannya, Blackbox testing digunakan untuk menguji berbagai aspek fungsional seperti validasi *input*, proses bisnis, serta *respon* sistem terhadap kondisi tertentu. Metode ini sangat efektif untuk menemukan kesalahan seperti fungsi yang tidak berjalan, kesalahan antarmuka, atau *output* yang tidak sesuai. Selain itu, Blackbox testing membantu memastikan bahwa sistem dapat digunakan dengan baik dari sudut pandang pengguna. Oleh karena itu, metode ini sering digunakan pada tahap akhir pengembangan perangkat lunak sebelum sistem dirilis ke publik.

2.10 Penelitian Terkait dan State-of-the-Art

Berbagai perguruan tinggi di Indonesia telah mulai mengadopsi asisten virtual untuk meningkatkan layanan informasi akademik mereka, dengan menerapkan pendekatan dan metode yang sangat beragam. Implementasi sistem ini mencakup berbagai platform populer, mulai dari *chatbot* yang terintegrasi di *web*, Telegram, WhatsApp, hingga aplikasi *mobile*. Metode pengembangan yang digunakan pun bervariasi, mulai dari pendekatan berbasis aturan (*rule-based*)

seperti *Artificial Intelligence Markup Language* (AIML) dan pencocokan pola (*pattern matching*) seperti *Fuzzy String Matching* dan TF-IDF, hingga pemanfaatan teknologi *deep learning* seperti *Recurrent Neural Network* (RNN). Berdasarkan analisis sistem yang sudah ada, mayoritas implementasi asisten virtual berfokus pada pemberian informasi pendaftaran mahasiswa baru, informasi akademik umum, serta layanan administratif dasar.

Evaluasi performa asisten virtual *existing* menunjukkan variasi signifikan dalam metrik akurasi, *response time*, dan *user satisfaction* berdasarkan teknologi yang diimplementasikan. Menurut Amrullah et al (2022), hasil pengujian *chatbot* untuk penerimaan mahasiswa baru pada Universitas Bumigora menggunakan metode *Black Box Testing* mendapatkan hasil 100% valid dengan *User Acceptance Testing* sebesar 94,67%. Penelitian lain menunjukkan bahwa *chatbot* menggunakan *Recurrent Neural Network* dapat menjawab pertanyaan mengenai pendaftaran mahasiswa baru dengan akurasi sebesar 88%, presisi sebesar 95%, dan *recall* sebesar 92%. *Benchmarking* juga mengungkapkan bahwa *response time* sistem berkisar antara 0,5-3 detik tergantung pada kompleksitas *query* dan arsitektur sistem, dengan sistem berbasis *cloud* umumnya menunjukkan performa yang lebih stabil dibandingkan *on-premise deployment*. *User satisfaction scores* menggunakan SUS menunjukkan rentang 68-85, mengindikasikan tingkat penerimaan pengguna yang bervariasi dari *acceptable* hingga *excellent*.

Analisis terhadap implementasi asisten virtual yang sudah ada mengidentifikasi beberapa kesenjangan yang sekaligus menjadi peluang pengembangan untuk menciptakan sistem yang lebih canggih. Meskipun asisten virtual telah banyak diterapkan, mayoritas sistem saat ini masih mengandalkan model *rule-based*, sehingga menunjukkan keterbatasan saat harus memberikan respons terhadap pertanyaan yang kompleks atau yang berada di luar basis pengetahuan yang telah ditetapkan. Kesenjangan utama yang teridentifikasi mencakup pemahaman bahasa alami untuk Bahasa Indonesia yang masih terbatas, minimnya personalisasi respons berdasarkan konteks pengguna, serta kurangnya integrasi antarmuka suara (*voice interface*) untuk meningkatkan aksesibilitas. Oleh karena itu, terdapat peluang signifikan untuk memanfaatkan LLM berbasis *generative AI* guna meningkatkan kemampuan sistem dalam memahami konteks

dan menghasilkan respons yang lebih natural. Selain itu, peluang juga terletak pada implementasi interaksi *multimodal* yang menggabungkan teks dan suara, serta integrasi tanpa celah dengan sistem informasi akademik yang sudah ada untuk menyediakan layanan nyata yang lebih komprehensif.

2.11 Kerangka Teoritis Penelitian

2.11.1 Technology Acceptance Model (TAM)

Technology Acceptance Model (TAM) merupakan kerangka kerja yang banyak digunakan untuk memahami penerimaan pengguna terhadap teknologi baru, termasuk asisten virtual. Faktor utama seperti *perceived usefulness* dan *perceived ease of use* berperan penting dalam menentukan sikap serta niat penggunaan teknologi (Octavia & Nugraha, 2024). Dalam konteks asisten virtual, penelitian menemukan bahwa aspek kepercayaan dan rasa menyenangkan turut memengaruhi persepsi pengguna terhadap kegunaan layanan (Zahro & Nugraha, 2021). Oleh karena itu, penerapan TAM pada asisten virtual harus mempertimbangkan konstruksi tambahan agar prediksi adopsi teknologi lebih komprehensif.

Selain konstruksi utama, faktor-faktor lain yang berpengaruh terhadap penerimaan teknologi meliputi kualitas sistem, privasi, dan keandalan respons. Studi tentang penerapan teknologi berbasis AI menunjukkan bahwa tingkat kepercayaan pengguna terhadap keamanan data memengaruhi sikap dan intensi perilaku (Octavia & Nugraha, 2024). Sementara itu, persepsi kenyamanan dan dukungan teknis terbukti menjadi determinan penting dalam keputusan pengguna untuk terus menggunakan layanan berbasis suara. Dengan demikian, model TAM yang diperluas dapat memberikan pemahaman lebih baik mengenai faktor penerimaan asisten virtual.

2.11.2 Human-Computer Interaction (HCI)

Prinsip desain interaksi suara dalam HCI menekankan pentingnya *naturalness*, latensi rendah, dan kejelasan umpan balik agar komunikasi terasa intuitif. Penelitian menunjukkan bahwa pengalaman pengguna meningkat ketika *voice assistant* mampu merespons perintah dengan cepat dan sesuai konteks (Kusmayadi et al., 2024). Evaluasi kegunaan melalui *usability testing* juga menjadi strategi penting untuk memastikan bahwa sistem dapat memenuhi ekspektasi

pengguna . Oleh sebab itu, desain interaksi suara perlu menggabungkan aspek teknis pengenalan bahasa dengan praktik desain berbasis pengalaman pengguna.

Prinsip *conversation design* menekankan pengelolaan dialog yang alami, jelas, dan sesuai tujuan percakapan. Penelitian tentang *chatbot* layanan publik menegaskan bahwa perancangan alur percakapan yang adaptif dan mudah dipahami meningkatkan kepuasan pengguna. Selain itu, kemampuan sistem dalam menangani kesalahan (*error handling*) menjadi indikator penting dalam menjaga kelancaran interaksi. Dengan demikian, penerapan *conversation design* membantu menciptakan percakapan yang lebih efektif dan menyerupai komunikasi manusia.

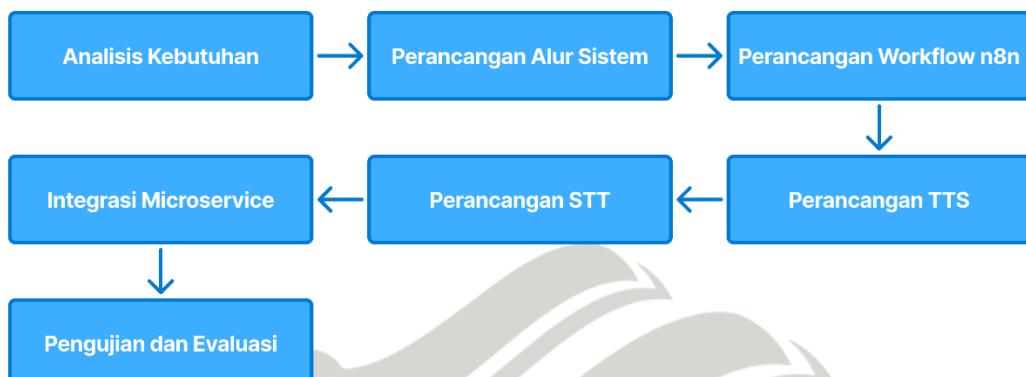
Pendekatan *user-centered design* (UCD) menekankan keterlibatan pengguna sejak tahap awal pengembangan, mulai dari analisis kebutuhan hingga uji coba prototipe. Studi pada *chatbot* pendidikan menunjukkan bahwa integrasi umpan balik langsung dari pengguna dapat meningkatkan akurasi percakapan dan penerimaan sistem. Dengan iterasi berulang, desain dapat disesuaikan secara dinamis agar benar-benar sesuai preferensi pengguna. Oleh karena itu, pendekatan UCD sangat relevan dalam pengembangan asisten virtual berbasis suara.

UNIVERSITAS
MA CHUNG

BAB III

Analisis dan Perancangan Sistem

3.1 Alur Penelitian



Gambar 3.1 Alur Penelitian

Proses penelitian dan pengembangan asisten virtual suara ini mengadopsi pendekatan *Agile* untuk memastikan sistem dapat dikembangkan secara fleksibel dan adaptif terhadap kebutuhan spesifik kampus. Alur dimulai dengan tahapan Analisis Kebutuhan yang berfokus pada identifikasi secara spesifik kebutuhan dan prosedur layanan akademik Universitas Ma Chung yang harus diakomodasi oleh sistem. Hasil analisis ini kemudian menjadi dasar untuk Perancangan Alur Sistem secara menyeluruh, yang menggambarkan bagaimana interaksi suara pengguna akan diproses dari awal hingga akhir. Dari alur sistem ini, dikembangkanlah Perancangan *workflow* n8n, yang berfungsi sebagai inti sistem untuk mengorkestrasi pemanggilan model LLM (Gemini 2.5 Flash) dan mekanisme *Retrieval-Augmented Generation* (RAG) ke *knowledge base*.

Setelah kerangka logika sistem (n8n) dirancang, proses berlanjut ke perancangan komponen suara, yaitu Perancangan TTS (*Text-to-Speech*) dan Perancangan STT (*Speech-to-Text*), yang menjamin konversi teks ke suara dan sebaliknya dapat berjalan dengan akurasi tinggi dan kualitas natural. Mengingat arsitektur yang digunakan adalah *microservice*, langkah-langkah selanjutnya berfokus pada integrasi *microservice*, di mana semua komponen yang dikembangkan (aplikasi web, n8n, STT, dan TTS) disatukan sebagai layanan yang terpisah namun saling terhubung. Seluruh sistem terintegrasi ini kemudian melalui

tahap Pengujian dan Evaluasi yang ketat untuk memastikan akurasi konten, kualitas suara, dan performa *end-to-end* sesuai dengan tujuan awal layanan akademik yang cepat dan efisien, sejalan dengan prinsip pengembangan iteratif.

3.2 Analisis Kebutuhan Sistem

Pengembangan asisten virtual untuk layanan akademik Universitas Ma Chung ini bertujuan untuk membantu mahasiswa dan civitas akademika dalam mengakses informasi administratif dan pengetahuan umum tentang universitas. Sistem ini tidak ditujukan untuk mendukung proses pembelajaran, melainkan fokus pada layanan informasi seperti prosedur administrasi akademik, jadwal perkuliahan, informasi fasilitas kampus, prosedur pendaftaran, dan informasi umum lainnya terkait Universitas Ma Chung. Sistem mengintegrasikan teknologi *Large Language Model* (LLM), *Text-to-Speech* (TTS), dan *Speech-to-Text* (STT) dalam arsitektur microservice untuk memberikan pengalaman interaksi yang natural dan responsif melalui input teks maupun suara.

Proses pengembangan sistem mengikuti alur yang sistematis dimulai dari analisis kebutuhan untuk mengidentifikasi kebutuhan fungsional dan non-fungsional sistem. Setelah kebutuhan teridentifikasi, dilakukan perancangan alur sistem yang menggambarkan interaksi antara pengguna dengan asisten virtual. Tahap selanjutnya adalah perancangan *workflow* n8n yang berfungsi sebagai *orchestrator* untuk mengintegrasikan berbagai komponen sistem termasuk Gemini LLM dengan implementasi RAG dan *vector database*. Perancangan kemudian dilanjutkan dengan arsitektur TTS menggunakan F5-TTS untuk menghasilkan *output* suara yang natural, serta implementasi STT untuk menerima *input* suara dari pengguna. Setelah semua komponen dirancang, dilakukan integrasi *microservice* yang menghubungkan aplikasi web, n8n *workflow*, dan F5-TTS melalui API dan *webhook*. Tahap akhir meliputi evaluasi performa setiap komponen sistem dan optimasi untuk meningkatkan kualitas respons, kecepatan pemrosesan, dan efisiensi *resource*.

Implementasi sistem memerlukan spesifikasi perangkat keras dan perangkat lunak yang memadai untuk menjalankan model AI dan memproses data multimedia. Tabel 3.1 menunjukkan spesifikasi perangkat keras yang digunakan dalam penelitian ini, sedangkan Tabel 3.2 menunjukkan spesifikasi perangkat keras yang

digunakan untuk evaluasi dan pada Tabel 3.3 menunjukkan daftar perangkat lunak dan tools yang diperlukan beserta fungsinya dalam mendukung pengembangan sistem.

Tabel 3.1 Komponen Perangkat Keras3;

Komponen	Spesifikasi	Fungsi
<i>Processor</i>	Intel Core i7-3820	Menjalankan <i>workflow</i> n8n, aplikasi web, dan proses komputasi umum
<i>Graphics Card</i>	NVIDIA GeForce RTX 4060 (8GB VRAM)	Mempercepat inferensi model LLM dan generasi audio TTS menggunakan CUDA
<i>Memory (RAM)</i>	32 GB	Menampung model AI dalam <i>memory</i> dan memproses data secara simultan
<i>Storage</i>	HDD dengan kapasitas kosong minimal 80 GB	Menyimpan model AI, <i>dataset</i> , <i>knowledge base</i> , dan <i>file</i> audio
<i>Microphone</i>	<i>Microphone</i> standar dengan sampling rate minimal 16kHz	Menangkap <i>input</i> suara pengguna untuk proses STT
<i>Speaker</i>	<i>Speaker</i> stereo standar	Memutar <i>output</i> audio hasil TTS kepada pengguna

Tabel 3.2 Komponen Perangkat Keras

Komponen	Spesifikasi	Fungsi
<i>Processor</i>	Intel Core i7-12700	Menjalankan <i>workflow</i> n8n, aplikasi web, dan proses komputasi umum
<i>Graphics Card</i>	NVIDIA GeForce RTX 3050 (8GB VRAM)	Mempercepat inferensi model LLM dan generasi audio TTS menggunakan CUDA
<i>Memory (RAM)</i>	16 GB	Menampung model AI dalam <i>memory</i> dan

Komponen	Spesifikasi	Fungsi
		memproses data secara simultan
<i>Storage</i>	HDD dengan kapasitas kosong minimal 50 GB	Menyimpan model AI, <i>dataset</i> , <i>knowledge base</i> , dan <i>file audio</i>
<i>Microphone</i>	<i>Microphone</i> standar dengan sampling rate minimal 16kHz	Menangkap <i>input</i> suara pengguna untuk proses STT
<i>Speaker</i>	<i>Speaker</i> stereo standar	Memutar <i>output</i> audio hasil TTS kepada pengguna

Tabel 3.3 Komponen Perangkat Lunak

Perangkat Lunak	Versi	Fungsi
PhpStorm	Latest	IDE untuk pengembangan aplikasi web berbasis PHP
Visual Studio Code	Latest	IDE untuk pengembangan F5-TTS
Docker Desktop	Latest	Platform <i>containerization</i> untuk menjalankan n8n dan isolasi <i>environment</i>
XAMPP	8.2.12	Local server untuk <i>development</i> aplikasi web PHP dengan Apache
Python	3.10.11	<i>Runtime environment</i> untuk menjalankan F5-TTS dan <i>library</i> AI
Git	2.51.0	<i>Version control system</i> untuk manajemen <i>source code</i>
Composer	6.3.0	<i>Dependency manager</i> untuk PHP
Node.js	22.19.0	<i>Runtime</i> untuk menjalankan n8n dan <i>tools</i> JavaScript
MySQL	8.0.43.0	<i>Database management system</i> untuk menyimpan data aplikasi
FFmpeg	8.0	<i>Library</i> untuk <i>processing</i> audio dan video
FFprobe	8.0	Tools untuk analisis file multimedia
Visual Studio Build Tools 2022	2022	<i>Compiler</i> dan <i>build tools</i> untuk kompilasi <i>library</i> Python yang memerlukan C++

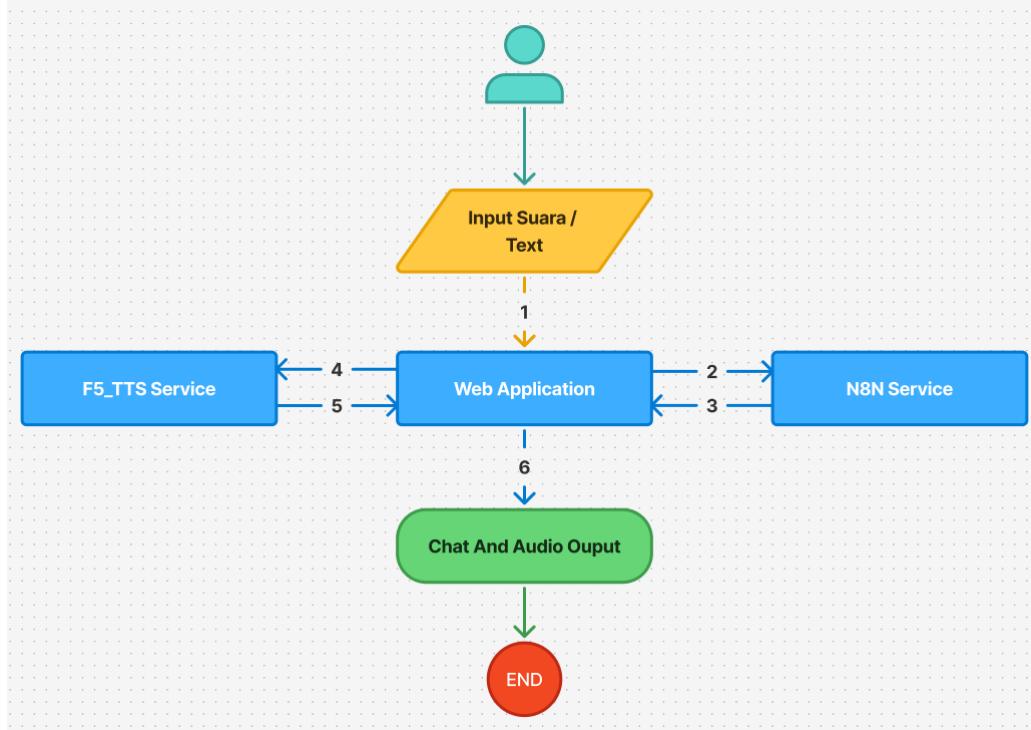
3.3 Perancangan Alur Sistem Asisten Virtual

Perancangan alur sistem asisten virtual ini dirancang untuk memastikan interaksi yang lancar dan responsif bagi pengguna (mahasiswa atau civitas akademik) yang ingin mendapatkan layanan administratif atau pengetahuan umum kampus. Sebelum interaksi pengguna dimulai, terdapat alur penting yang melibatkan pengelola *knowledge base*. Pihak universitas atau pengembang bertanggung jawab untuk memasukkan materi pengetahuan berupa dokumen-dokumen akademik dan administratif ke dalam sistem melalui antarmuka aplikasi web. Dokumen yang diunggah ini kemudian akan dikirimkan ke layanan n8n untuk diproses. Tujuan utama dari proses ini adalah mengubah dokumen mentah menjadi *knowledge base* yang terstruktur dan siap digunakan, memastikan bahwa sistem selalu memiliki informasi yang paling akurat dan terkini.

Proses selanjutnya berfokus pada penyimpanan dan persiapan data. Setelah dokumen diterima oleh n8n, ia akan menjalankan serangkaian tahapan seperti pemecahan teks (*chunking*) dan pembuatan representasi numerik (*embedding*) dari setiap bagian dokumen. *Embedding* ini kemudian akan disimpan di dalam *Vector Database* sebagai *knowledge base* sistem. Langkah ini sangat krusial karena basis data vektor memungkinkan sistem untuk melakukan pencarian semantik yang cepat dan akurat, di mana sistem dapat mencocokkan pertanyaan pengguna dengan *knowledge* yang paling relevan, bukan hanya berdasarkan kata kunci. Dengan basis pengetahuan yang sudah terpasang, sistem siap menerima dan memproses permintaan dari pengguna.

Interaksi pengguna diawali ketika pengguna berinteraksi melalui *input* suara langsung atau *chat* teks yang semuanya diakomodasi melalui *interface* dan *backend* aplikasi web berbasis PHP. Permintaan yang diterima oleh aplikasi web segera dilempar ke *service* n8n yang berjalan di lingkungan Docker melalui *webhook*. Di dalam n8n, pemrosesan informasi dilakukan secara cerdas dengan memanfaatkan model *Large Language Model* (LLM) dari Gemini 2.5 Flash untuk mengekstrak konteks permintaan atau pertanyaan, yang kemudian dikombinasikan dengan teknik *Retrieval-Augmented Generation* (RAG) untuk mendapatkan *knowledge* yang dibutuhkan dari *Vector Database*. Setelah jawaban dikembalikan ke aplikasi web PHP, jawaban tersebut dikirimkan ke *endpoint service* F5-TTS untuk diubah

menjadi audio secara per *chunk*. Setiap *chunk* audio yang selesai digenerasi akan segera dikirimkan kembali ke aplikasi web PHP dan langsung diputar suaranya, sehingga pengalaman interaksi suara menjadi cepat dan dinamis.



Gambar 3.2 Alur Sistem Asisten Virtual

3.4 Arsitektur Web Application

Aplikasi web untuk asisten virtual ini dirancang menggunakan arsitektur *Model-View-Controller* (MVC). Pola arsitektur ini dipilih secara spesifik untuk memastikan kode program berjalan secara terstruktur, modular, dan memfasilitasi *maintenance* program yang lebih mudah di masa depan. Dalam kerangka MVC, setiap permintaan atau interaksi dari pengguna akan diterima dan diarahkan pertama kali oleh *Controller*. *Controller* ini berfungsi sebagai jembatan yang menerima *input*, memanggil logika bisnis yang diperlukan dari *Service*, dan akhirnya memilih *View* yang tepat untuk menyajikan hasil respons kembali kepada pengguna. Pemisahan tanggung jawab ini memungkinkan pengembang untuk mengelola logika aplikasi secara independen dari presentasi antarmuka, yang merupakan kunci efisiensi dalam pengembangan sistem yang kompleks.

Selain itu, arsitektur *backend* web aplikasi ini memanfaatkan *Object-Relational Mapping* (ORM) untuk berinteraksi dengan MySQL, yang digunakan sebagai *database* relasional utama. MySQL dipilih untuk menyimpan data

pengguna dan data sesi yang diperlukan untuk fungsionalitas asisten virtual. ORM digunakan untuk mengelola dan memanipulasi data ini secara efisien, memfasilitasi interaksi dengan *database* tanpa perlu menulis *query SQL* secara langsung. ID unik dari data sesi atau pengguna yang disimpan di MySQL ini sangat penting karena akan dikirimkan bersama permintaan pengguna ke *service n8n*. Tujuannya adalah agar n8n dapat mengingat konteks percakapan lampau pengguna tersebut. Dengan meneruskan ID sesi yang *persistent*, sistem dapat menghindari kondisi *lost memory* dan memberikan jawaban yang relevan berdasarkan riwayat perbincangan, sehingga menciptakan pengalaman interaksi yang kohesif dan personal.

Peran kedua dari database MySQL adalah untuk mendukung manajemen otorisasi dan pemisahan peran (*role*) dalam sistem. Data yang disimpan di MySQL digunakan untuk membedakan antara sesi pengunjung, pengguna terdaftar (mahasiswa/civitas), dan administrator. Pemisahan ini sangat krusial untuk aspek keamanan, terutama untuk membatasi akses ke halaman administrasi yang digunakan untuk konfigurasi asisten virtual dan pengelolaan *knowledge base*. Dengan verifikasi peran pengguna melalui *database*, sistem dapat menjamin bahwa halaman sensitif atau fitur konfigurasi *backend* tidak dapat diakses oleh publik atau pengguna yang tidak memiliki otorisasi (*unauthorized access*), menjaga integritas dan keamanan seluruh sistem asisten virtual.

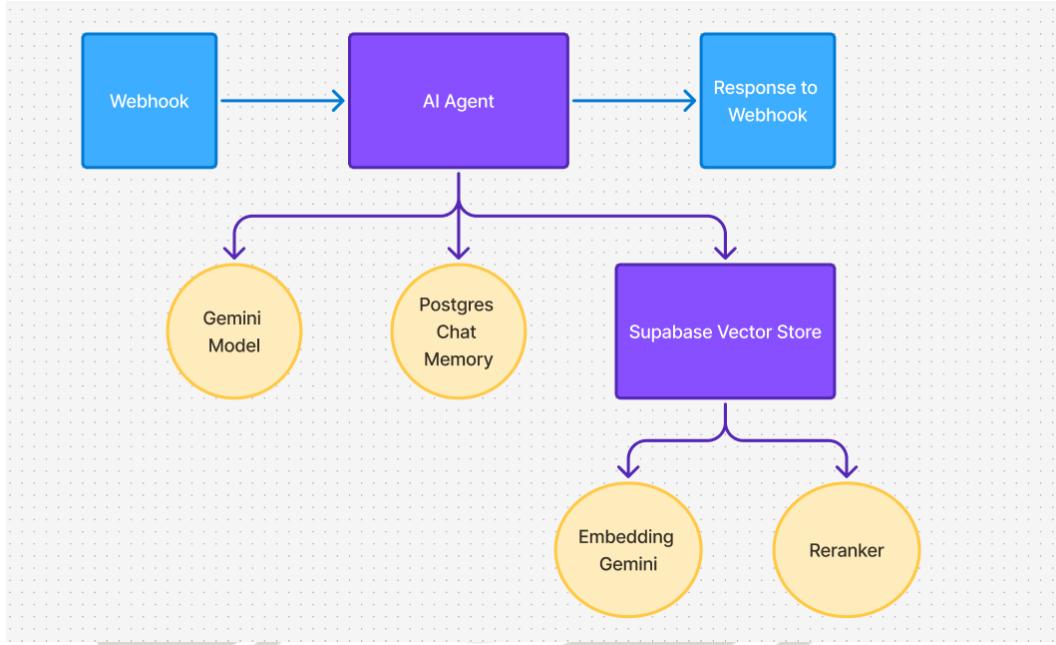
3.5 Perancangan Arsitektur LLM dengan n8n

3.5.1 Implementasi n8n pada Docker

Untuk menjamin fleksibilitas, skalabilitas, dan kemudahan dalam *deployment* layanan, platform n8n akan diimplementasikan menggunakan teknologi Docker. Penggunaan Docker memungkinkan n8n untuk berjalan secara terisolasi dan konsisten di berbagai lingkungan server, menghindari isu dependensi, dan mempermudah proses *maintenance* dan *update*. Layanan n8n yang berjalan di dalam *container* Docker ini akan dipublikasikan ke jaringan eksternal melalui Port 5678. Semua permintaan yang berasal dari *backend* aplikasi web PHP, khususnya *payload* pertanyaan dari pengguna, akan diarahkan sebagai tembakan API *webhook* ke alamat server yang spesifik dengan menggunakan Port 5678 ini. Arsitektur ini memastikan bahwa komunikasi antara aplikasi web dan mesin pemroses logika

(*logic processing engine*) n8n dapat berjalan dengan cepat dan aman, memanfaatkan *webhook* sebagai titik integrasi utama.

3.5.2 Perancangan Workflow n8n



Gambar 3.3 Rancangan Workflow n8n

Perancangan *workflow* pada n8n merupakan inti dari sistem pemrosesan permintaan asisten virtual, yang dimulai dengan penerimaan *payload* data dari aplikasi web. *Workflow* akan diawali oleh *node webhook* yang bertugas menerima data pertanyaan pengguna, termasuk ID sesi yang krusial untuk menjaga konteks percakapan. Setelah diterima, *payload* akan langsung diteruskan ke *node AI Agent*, yang berfungsi sebagai orkestrator utama dalam menentukan langkah pemrosesan selanjutnya. *AI Agent* ini dikonfigurasi untuk terhubung dengan LLM Gemini 2.5 Flash sebagai mesin utama pembuat jawaban yang natural. Untuk memastikan jawaban yang akurat, *agent* ini terintegrasi dengan dua layanan krusial dari Supabase, yang pertama adalah PostgreSQL untuk menyimpan riwayat (*history*) chat pengguna, yang digunakan untuk mempertahankan memori percakapan dan yang kedua adalah layanan *Retrieval-Augmented Generation* (RAG) dari Supabase, yang bertugas mengambil *knowledge* relevan dari dokumen kampus yang sudah diunggah.

Untuk meningkatkan kualitas dan akurasi jawaban yang didapatkan dari proses RAG, *workflow* ini mengimplementasikan *reranker* setelah proses

pengambilan dokumen. Reranker berfungsi untuk memfilter dan memberi peringkat ulang potongan *knowledge* yang paling relevan sebelum disajikan kepada LLM, sehingga model hanya berfokus pada informasi yang paling tepat dan menghindari *hallucination*. Setelah AI *Agent* berhasil merangkum jawaban yang akurat dan kontekstual, jawaban tersebut akan dikembalikan ke aplikasi web melalui *node Response to Webhook*. *Node* ini menutup siklus pemrosesan di n8n dengan mengirimkan hasil akhir berupa teks jawaban kembali ke *backend PHP*, yang kemudian akan diproses lebih lanjut untuk konversi menjadi audio.

3.5.3 Integrasi LLM Gemini 2.5 Flash Pada n8n

Integrasi model bahasa besar Gemini 2.5 Flash ke dalam *workflow* n8n merupakan langkah fundamental untuk mengaktifkan kapabilitas pemrosesan bahasa alami dan pembuatan jawaban yang cerdas. Integrasi ini dilakukan dengan memanfaatkan *node* Gemini yang tersedia di dalam platform n8n. Langkah setup dimulai dengan memasukkan API Key Gemini yang telah didapatkan dari Google AI Studio ke dalam konfigurasi *node*. Kunci API ini berfungsi sebagai *otentikasi* agar n8n dapat mengirimkan permintaan dan menerima respons dari Google. Setelah *otentikasi* berhasil, model yang spesifik, yaitu Gemini 2.5 Flash, dipilih dalam pengaturan *node*. Pemilihan model ini didasarkan pada kebutuhan sistem yang memerlukan model yang sangat cepat dan efisien untuk interaksi layanan akademik waktu nyata, sambil tetap mempertahankan kualitas *reasoning* yang tinggi untuk tugas RAG.

3.5.4 Implementasi Retrieval-Augmented Generation (RAG)

Implementasi *Retrieval-Augmented Generation* (RAG) pada *workflow* n8n bertujuan untuk memastikan asisten virtual memberikan jawaban yang faktual dan spesifik berdasarkan dokumen-dokumen resmi Universitas Ma Chung. Layanan RAG ini diintegrasikan melalui Supabase, yang berfungsi sebagai basis data vektor penyimpan *knowledge* kampus. Langkah awal implementasi adalah menghubungkan n8n dengan akun Supabase, yang memerlukan proses otentikasi dan *login*. Setelah berhasil *login*, n8n akan mendapatkan alamat kredensial spesifik yang digunakan untuk menjalin koneksi aman dan berkelanjutan dengan layanan *database Supabase*.

Dalam *workflow* n8n, integrasi RAG dilakukan dengan menggunakan *node Retrieve Document*. *Node* ini dikonfigurasi untuk menjalankan fungsi RAG, di mana ia akan menerima *query* dari pengguna, mencarinya di basis data vektor Supabase, dan mengambil potongan dokumen yang paling relevan. Kunci dari akurasi sistem ini adalah penambahan aturan (*rule*) atau instruksi pada *prompt* yang dikirimkan ke LLM Gemini 2.5 Flash. *Rule* utama yang diterapkan adalah membatasi LLM untuk hanya mengambil dan merangkum informasi dari dokumen yang tersedia (yang diambil oleh RAG). Selain itu, *rule* lain juga dapat ditambahkan, seperti panduan gaya bahasa, format jawaban, atau instruksi untuk menjaga tindakan LLM agar selalu sesuai dengan konteks akademik dan administratif kampus, memastikan respons yang aman dan profesional.

3.5.5 Implementasi Vector Database

Implementasi *Vector Database* menjadi elemen vital dalam arsitektur RAG, namun untuk fungsi manajemen memori asisten virtual, sistem memanfaatkan koneksi database relasional yang terintegrasi dengan layanan yang sama. Untuk tujuan ini, sistem menggunakan PostgreSQL yang *di-host* melalui platform Supabase. Koneksi ke layanan *database* ini dikelola di dalam *workflow* n8n melalui *node Database* khusus untuk PostgreSQL. *Node* ini tidak berfungsi untuk penyimpanan *vector embedding* dokumen kampus yang ditangani oleh *node RAG*, melainkan secara spesifik digunakan untuk mengelola riwayat (*history*) *chat* dari setiap sesi atau pengguna yang berinteraksi. Penggunaan *database* relasional untuk penyimpanan riwayat ini memungkinkan sistem untuk mempertahankan ID sesi yang *persisten*, sehingga setiap kali pengguna mengajukan pertanyaan baru, n8n dapat merujuk kembali pada riwayat percakapan yang tersimpan di PostgreSQL. Mekanisme ini memastikan bahwa asisten virtual memiliki memori kontekstual dan dapat memberikan respons yang kohesif serta berkelanjutan, layaknya sebuah percakapan alami.

3.5.6 Pemberian Knowledge Base pada Model

Pemberian *knowledge base* kepada model merupakan langkah terpisah yang memastikan asisten virtual memiliki akses terhadap seluruh dokumen akademik dan administratif kampus. Proses ini tidak terjadi dalam alur tanya jawab pengguna, melainkan melalui *workflow* tersendiri di n8n yang didedikasikan hanya untuk

fungsi *document ingestion* (pemasukan dokumen). *Workflow* ini dirancang untuk menerima dokumen-dokumen resmi yang diunggah oleh pihak universitas melalui aplikasi web. Setelah dokumen diterima oleh *workflow ingestion*, dokumen akan langsung diproses dan dikirimkan ke *node* Supabase yang bertugas untuk fungsi *insert* dokumen. *Node* ini akan menjalankan proses pra-pemrosesan data, termasuk pemecahan dokumen menjadi *chunk* kecil dan pembuatan *vector embedding* dari setiap *chunk* tersebut. *Vector embedding* inilah yang kemudian disimpan ke dalam *Vector Database* di Supabase, menjadikannya siap untuk digunakan oleh mekanisme *Retrieval-Augmented Generation* (RAG) dalam *workflow* utama. Proses yang terisolasi ini memastikan bahwa pembaruan *knowledge base* dapat dilakukan secara efisien tanpa mengganggu performa dan ketersediaan layanan interaksi asisten virtual.

3.5.7 Konfigurasi Webhook

Konfigurasi *Webhook* merupakan tahap esensial karena ia berfungsi sebagai gerbang utama (*gateway*) untuk komunikasi data dua arah antara aplikasi web PHP dengan *workflow* pemrosesan di n8n. *Webhook* ini memiliki peran ganda. Pertama, sebagai gerbang masuk yang siap menerima payload data pertanyaan dari pengguna yang dikirim oleh *backend* aplikasi web dan yang kedua, sebagai gerbang respons yang akan mengembalikan jawaban akhir dari n8n ke aplikasi web. Dalam implementasinya, *Webhook* ini akan memiliki URL *Webhook* yang unik. URL ini, yang merujuk pada alamat *production* n8n pada Docker, akan disimpan pada file .env di dalam konfigurasi aplikasi web. Setiap kali aplikasi web ingin mengirimkan permintaan ke asisten virtual, ia akan menargetkan URL ini menggunakan metode HTTP POST yang aman dan sesuai untuk pengiriman *payload* data, memastikan alur data pertanyaan dan respons berjalan secara terstruktur dan efisien.

3.6 Perancangan Arsitektur Text-to-Speech (F5-TTS)

3.6.1 Implementasi F5-TTS dan Chatterbox Menggunakan Virtual Environment Python

Layanan F5-Text-to-Speech (F5-TTS) dan Chatterbox merupakan komponen krusial untuk mengonversi jawaban teks menjadi *output* suara yang natural dan akan diimplementasikan menggunakan lingkungan Python. Untuk

memastikan stabilitas, keterpisahan dependensi, dan kemudahan pengelolaan *library* yang dibutuhkan, F5-TTS akan dijalankan di dalam *Virtual Environment* (venv) Python. Penggunaan venv sangat penting karena ia menciptakan lingkungan yang terisolasi dari global interpreter Python. Isolasi ini mencegah konflik versi *library* dengan aplikasi lain di server dan memastikan bahwa F5-TTS dan Chatterbox selalu berjalan dengan *dependency* spesifik yang telah diuji dan diperlukan, seperti pustaka untuk pemrosesan audio dan model sintesis suara. Lingkungan virtual ini akan menampung semua *library* yang diperlukan untuk layanan TTS, dan layanan ini akan dipublikasikan sebagai *endpoint API* yang akan diakses oleh aplikasi web PHP untuk permintaan konversi teks-ke-suara.

3.6.2 Penyediaan Endpoint F5-TTS dan Chatterbox

Karena *service* F5-Text-to-Speech (F5-TTS) dan Chatterbox tidak secara bawaan menyediakan *endpoint API* yang siap diakses, diperlukan pembangunan *endpoint custom* sendiri pada *port* yang berbeda untuk menjembatani komunikasi dengan aplikasi web PHP. Tugas ini melibatkan pembuatan *code* program menggunakan framework Python FastAPI, yang akan menjadi *wrapper* bagi fungsi inti F5-TTS. *Code custom* baru ini akan diekspos sebagai *endpoint* HTTP POST. Ketika aplikasi web PHP mengirimkan teks jawaban ke *endpoint* ini, *endpoint* akan memicu fungsi *inference* F5-TTS dan Chatterbox yang sudah tersedia di lingkungan virtual Python. *Inference* tersebut akan menghasilkan *output* audio, yang kemudian akan dikirimkan kembali ke aplikasi web. Pembuatan *endpoint custom* ini memastikan bahwa fungsi canggih dari F5-TTS dapat dimanfaatkan secara efisien dan terstruktur dalam alur komunikasi sistem keseluruhan.

3.6.3 Konfigurasi Referensi Audio dan Referensi Teks

Agar layanan F5-TTS mampu menghasilkan suara *output* yang optimal dan natural, sistem memerlukan referensi audio yang akan dijadikan suara duplikat (*voice clone*) untuk sintesis. F5-TTS akan menggunakan karakteristik suara dari referensi audio ini sebagai dasar untuk suara yang dihasilkan oleh asisten virtual. Selain itu, untuk memastikan proses konversi Teks-ke-Suara (*Text-to-Speech*) berjalan dengan akurat, *endpoint* F5-TTS juga membutuhkan referensi teks yang sesuai dengan isi dari referensi audio tersebut. Pasangan referensi audio dan teks ini penting untuk memvalidasi dan mengoptimalkan model *cloning*, sehingga hasil

output TTS memiliki intonasi dan pelafalan yang tepat. Aspek yang paling fleksibel dari konfigurasi ini adalah bahwa kedua referensi ini akan dikirimkan dari sisi aplikasi web PHP bersamaan dengan teks yang akan dikonversi. Mekanisme pengiriman dinamis ini memungkinkan pengembang atau administrator untuk mengubah jenis suara asisten virtual sesuai kebutuhan (*on-the-fly*), memberikan kontrol penuh terhadap identitas suara asisten virtual.

3.6.4 Optimasi Generatif Audio

Optimasi pada proses generatif audio sangat penting untuk mengatasi isu latensi tinggi yang terjadi pada implementasi F5-TTS bawaan, di mana sistem akan menghasilkan seluruh *batch* teks dalam satu waktu sebelum mengirimkannya ke aplikasi web. Pada saat text yang akan diubah menjadi suara ini terlalu panjang akan membuat proses *generate voice* sangat lama. Proses yang terlalu lama ini dapat menyebabkan pengalaman pengguna yang buruk dan berpotensi membuat pengguna meninggalkan asisten virtual sebelum asisten virtual memberikan response. Untuk mengatasi hal ini, dilakukan modifikasi sistem dengan mengimplementasikan mekanisme pemotongan teks (*chunking*). Teks jawaban yang diterima dari n8n akan dibagi menjadi beberapa *chunk* kecil sebelum dikirimkan ke *endpoint* F5-TTS.

Setelah *chunking*, F5-TTS akan memproses dan menggenerasi audio untuk setiap *chunk* secara berurutan. Kunci dari optimasi ini terletak pada pengiriman respons. Setiap *chunk* audio yang selesai digenerasi akan segera dikirim kembali ke aplikasi web PHP tanpa menunggu *chunk* lainnya selesai. Untuk memfasilitasi pengiriman data *streaming* secara *real-time* ini, aplikasi web PHP memerlukan implementasi *WebSocket*. *WebSocket* memungkinkan koneksi dua arah yang *persisten*, sehingga setiap *chunk* audio dapat langsung *di-play* oleh browser pengguna sesaat setelah diterima. Dengan metode *chunking* dan *WebSocket* ini, asisten virtual dapat menghasilkan suara dengan cepat dan responsif, secara signifikan mengurangi waktu tunggu pengguna.

3.7 Implementasi Speech-to-Text (STT)

Implementasi fitur *Speech-to-Text* (STT) untuk mengubah input suara pengguna menjadi teks dilakukan sepenuhnya di sisi klien (*client-side*). Pendekatan ini dipilih untuk memaksimalkan kecepatan dan efisiensi, serta mengurangi beban

pemrosesan pada server aplikasi. Proses STT ini memanfaatkan kapabilitas *native* yang sudah tersedia pada browser modern, di mana fungsionalitasnya diakses langsung melalui JavaScript. Penggunaan API browser ini memungkinkan sistem untuk langsung merekam dan mengonversi ucapan pengguna menjadi teks secara *real-time*. Teks hasil konversi inilah yang kemudian akan dikirimkan sebagai *payload* melalui aplikasi web PHP ke *workflow* n8n, memungkinkan interaksi suara yang cepat dan responsif tanpa perlu layanan STT eksternal yang kompleks.

3.8 Integrasi Arsitektur Microservice

3.8.1 Integrasi Web Application dengan n8n

Integrasi antara Aplikasi Web PHP dengan *workflow* pemrosesan di n8n dilakukan melalui mekanisme *Webhook* untuk memastikan komunikasi yang cepat dan asinkron. URL *Webhook* yang unik dari n8n disimpan pada file .env dari sisi aplikasi web, memungkinkannya untuk menargetkan layanan n8n secara langsung. Ketika pengguna memberikan *input* (berbicara atau chat), aplikasi web akan menggunakan AJAX (Asynchronous JavaScript and XML) untuk memastikan proses berjalan secara *async*. *Input* tersebut tidak langsung dikirim ke n8n, melainkan diarahkan terlebih dahulu ke *endpoint* internal pada aplikasi web itu sendiri. Pada *endpoint* ini, *Controller* yang telah diatur akan mengambil alih, memproses *payload* dan memicu Service yang bertugas mengirimkan payload tersebut ke *Webhook* n8n. Pendekatan ini memastikan bahwa proses *backend* ke n8n tidak menghambat kinerja antarmuka pengguna, dan logika pengiriman data dapat dikelola secara terpusat oleh pola MVC.

3.8.2 Integrasi Web Application dengan F5-TTS

Integrasi antara Aplikasi Web PHP dengan layanan F5-TTS dimulai segera setelah *service* web aplikasi menerima teks jawaban dari *workflow* n8n. Respons teks ini akan dikembalikan ke *Controller* yang telah menangani permintaan awal. *Controller* kemudian akan langsung mengirimkan hasil *response* n8n ke *endpoint* dari F5-TTS. Pada proses ini teks *response* n8n akan dikirim bersama dengan referensi audio dan teks refrensi audio yang diperlukan ke *endpoint* F5-TTS yang sudah dibuat dan kemudian akan berjalan di *Virtual Environment* Python. Untuk mengoptimalkan kecepatan respons, proses konversi teks-ke-suara pada F5-TTS

dilakukan secara *chunking*. Setiap *chunk* audio yang selesai digenerasi akan segera dikirimkan kembali ke aplikasi web. Untuk menerima dan memutar *streaming* audio *real-time* ini, *frontend* aplikasi web memanfaatkan koneksi *WebSocket* yang *persisten*. Dengan cara ini, suara asisten virtual dapat langsung dimainkan kepada pengguna tanpa jeda yang signifikan, menciptakan pengalaman interaksi suara yang cepat dan *real-time*.

3.9 Mekanisme Fallback

Mekanisme *fallback* yang dirancang untuk asisten virtual Anda sangat penting untuk menjamin ketersediaan dan pengalaman pengguna yang konsisten, bahkan saat terjadi kegagalan sistem. Pendekatan utamanya adalah implementasi blok *try-catch* di setiap proses *service*. Ketika service web *application* utama mati atau gagal, blok *catch* akan langsung mengalihkan pengguna ke halaman khusus yang menginformasikan bahwa web sedang dalam *maintenance*. Jika terjadi *error* pada core asisten virtual yang dijalankan oleh n8n maka halaman interaksi akan menampilkan notifikasi atau pemberitahuan yang memberitahu pengguna bahwa aplikasi sedang dalam perbaikan. Strategi ini memastikan bahwa pengguna tidak dibiarkan tanpa respons atau mengalami crash aplikasi.

Selanjutnya, mekanisme *fallback* juga secara spesifik menangani kegagalan pada fitur-fitur pendukung, seperti *Text-to-Speech* (TTS) dan *Speech-to-Text* (STT). Jika sistem TTS mengalami *error* maka blok *catch* akan memicu tampilan pesan bahwa fitur suara sedang dalam proses *maintenance*. Namun, kegagalan pada TTS ini tidak akan menghentikan operasi inti asisten virtual. Pengguna akan tetap dapat berinteraksi penuh dengan asisten, menggunakan teks *chat*. Demikian juga pada sistem STT jika mengalami kegagalan, pengguna akan diberikan pesan perbaikan dan tetap dapat berinteraksi dengan *chat*. Desain bertujuan untuk meminimalkan sistem mati hingga tidak dapat digunakan sama sekali.

3.10 Pengujian dan Evaluasi Rancangan Sistem

Proses pengujian yang dilakukan pada penelitian ini adalah dengan mengujikan asisten virtual kepada 30 mahasiswa Universitas Ma Chung. Mahasiswa melakukan percobaan dengan memberikan pertanyaan langsung kepada asisten virtual dengan pertanyaan tentang Universitas Ma Chung. Setelah mahasiswa selesai mencoba maka akan mengisi microsoft form untuk melakukan

evaluasi. Evaluasi tahap pertama adalah evaluasi SUS dengan 10 pertanyaan kemudian berlanjut pada evaluasi Generatif AI dengan *Likert Scale Evaluation* yang terdapat 5 pertanyaan, kemudian berlanjut pada evaluasi kualitas TTS dengan menggunakan *Mean Opinion Score* sebanyak 5 pertanyaan dan terakhir melakukan evaluasi *Word Error Rate* dengan memasukkan total keseluruhan kata, total kata salah dan total kata hilang dari semua pertanyaan yang diajukan setiap mahasiswa. Mengingat keterbatasan dalam membawa perangkat PC pribadi, maka pengujian akan dilakukan dengan cara menjalankan asisten virtual pada komputer laboratorium Universitas Ma Chung dengan spesifikasi yang sudah memenuhi spesifikasi minimum

3.10.1 Pengujian dan Evaluasi Non-Fungsionalitas

Pengujian non-fungsionalitas juga sangat penting mengingat arsitektur sistem asisten virtual ini dibangun di atas konsep *microservice*, di mana aplikasi web PHP, n8n, dan F5-TTS berjalan sebagai layanan yang terpisah. Tahap pengujian ini untuk memastikan integrasi antar layanan dapat berjalan secara sempurna dan *end-to-end*. Pengujian ini akan melibatkan 30 responden menggunakan metode *System Usability Scale* (SUS) untuk mengukur tingkat penerimaan dan kemudahan penggunaan sistem.

Pertanyaan SUS akan berjumlah 10 pertanyaan yang setiap pertanyaannya akan berisikan jawaban 1 yang mewakili sangat tidak setuju dan 5 yang mewakili sangat setuju. Proses perhitungan melibatkan dua langkah kunci, untuk item bernomor ganjil sebagai pernyataan positif, skor yang didapatkan adalah hasil pengurangan nilai respons dengan angka 1. Lalu untuk item bernomor genap sebagai pernyataan negatif, skor diperoleh dari pengurangan angka 5 dengan nilai respons. Hasil jumlah skor 10 pertanyaan setiap responden akan dikalikan dengan 2,5 dan untuk mendapatkan hasil akhirnya adalah dengan melakukan rata-rata skor yang sudah dikali dengan 2,5. Daftar pernyataan SUS yang akan diberikan adalah sebagai berikut.

Tabel 3.4 Tabel Pernyataan *System Usability Scale*

No.	Pernyataan <i>System Usability Scale</i>	Tipe
1	Saya berpikir akan sering menggunakan Asisten Virtual Layanan Akademik ini.	Positif
2	Saya merasa Asisten Virtual ini terlalu rumit untuk digunakan.	Negatif
3	Saya merasa Asisten Virtual ini mudah untuk digunakan.	Positif
4	Saya rasa saya memerlukan bantuan dari staf akademik/teknisi untuk menggunakan Asisten Virtual ini.	Negatif
5	Saya merasa fitur-fitur yang ada di Asisten Virtual ini terhubung dan terintegrasi dengan baik.	Positif
6	Saya merasa ada banyak hal yang tidak konsisten/tidak serasi dalam Asisten Virtual ini.	Negatif
7	Saya yakin pengguna baru dapat belajar menggunakan Asisten Virtual ini dengan sangat cepat.	Positif
8	Saya merasa sangat merepotkan untuk menggunakan Asisten Virtual ini.	Negatif
9	Saya merasa percaya diri dan tidak khawatir membuat kesalahan saat menggunakan Asisten Virtual ini.	Positif
10	Saya perlu mempelajari banyak hal sebelum saya bisa menggunakan Asisten Virtual ini secara efektif.	Negatif

3.10.2 Pengujian dan Evaluasi Hasil Generatif AI

Untuk menjamin efektivitas Asisten Virtual sebagai layanan akademik, seluruh hasil generatif dari AI Agent (n8n dan Gemini 2.5 Flash) akan diuji dengan metode *blackbox testing* dengan matriks *Likert Scale Evaluation* yang akan dievaluasi oleh responden. Penggunaan matriks ini karena asisten virtual ditujukan untuk dapat menjawab semua pertanyaan pengguna berdasarkan informasi yang sudah ditanamkan pada asisten virtual. Pengguna yang menjadi tolak ukur penilaian apakah jawaban yang diberikan sudah menjawab dengan jelas dan sesuai dengan harapan mereka. Pertanyaan *Likert Scale Evaluation* terdapat 5 pertanyaan yang akan dijawab berupa *rating* 1 hingga 5 yang mana 1 mempresentasikan sangat buruk dan 5 sangat baik. Cara menghitungnya akan mengikuti rumus umum dari

Likert yaitu total skor dibagi dengan jumlah responden. Pertanyaan *Likert* yang akan digunakan adalah sebagai berikut.

Tabel 3.5 Pertanyaan *Likert Scale Evaluation*

No.	Pertanyaan Untuk <i>Rating Likert Scale Evaluation</i>
1	Seberapa akurat jawaban yang diberikan?
2	Seberapa relevan jawaban yang diberikan?
3	Seberapa lengkap jawaban yang diberikan?
4	Seberapa jelas jawaban yang diberikan?
5	Seberapa puas atas jawaban yang diberikan?

3.10.3 Pengujian dan Evaluasi Hasil Suara TTS

Pengujian dan evaluasi terhadap hasil *Text-to-Speech* (TTS) dari F5-TTS akan menggunakan *blackbox testing* dengan matriks pengujian menggunakan *Mean Opinion Score* (MOS). Matriks ini dipilih karena setiap orang dapat berbeda-beda kemampuan dalam mendengarkan pengucapan suatu bahasa. Hal itu bisa dipengaruhi oleh banyak faktor seperti *mother language*, aksen bahasa tiap daerah dan kemampuan alami pendengaran. Terdapat 5 pertanyaan MOS yang akan dijawab dengan jawaban berbentuk *rating* 1 hingga 5 yang mana 1 mempresentasikan sangat buruk dan 5 sangat baik. Cara menghitungnya akan mengikuti rumus umum dari MOS yaitu total skor dibagi dengan jumlah responden. Pertanyaan MOS yang akan digunakan adalah sebagai berikut.

Tabel 3.6 Pernyataan *Mean Opinion Score*

No.	Pertanyaan Untuk <i>Mean Opinion Score</i>
1	Seberapa natural suara yang dihasilkan?
2	Seberapa jelas suara yang dihasilkan?
3	Seberapa baik pelafalan suara yang dihasilkan?
4	Seberapa baik intonasi suara yang dihasilkan?
5	Seberapa baik kecepatan suara yang dihasilkan?

3.10.4 Pengujian dan Evaluasi Hasil Text STT

Evaluasi terhadap hasil *Speech-to-Text* (STT) berfokus pada akurasi konversi teks dan kemampuan sistem menangani variasi *input* ucapan pengguna, yang dapat diukur menggunakan metrik *Word Error Rate* (WER). Kriteria utama pengujian adalah memastikan kesesuaian antara suara yang diucapkan dengan teks yang dihasilkan melalui sebuah simulasi. Akurasi yang tinggi pada konversi STT ini sangat penting, karena *output* teks inilah yang menjadi *input* utama bagi *workflow* n8n. Jika teks awal salah, seluruh proses pencarian dan pemberian jawaban akan terganggu, sehingga pengujian akurasi STT menjadi hal yang sangat penting. Cara pengambilan data evaluasinya adalah dengan memasang *logger* pada asisten virtual. Ketika pengujian selesai dilakukan dan semua evaluasi sudah diselesaikan, maka *logger* pada *console* web akan dibuka dan akan dihitung langsung total kata dari pertanyaan yang diajukan, total kata salah dan total kata hilang yang divalidasi langsung oleh responden yang bersangkutan. Data tersebut akan langsung diisikan pada *form* evaluasi dan akan di-submit oleh responden. Setelah semua data terkumpul baru dilakukan perhitungan dengan rumus umum WER.

3.10.5 Pengujian Perbandingan Akurasi dan Latensi Asisten Virtual dengan Gemini

Pada evaluasi ini akan dilakukan perbandingan untuk menilai akurasi dan latensi jawaban antara asisten virtual dan Gemini. Untuk menjaga keseimbangan pengujian, dokumen akan diunggah satu per satu, baik pada asisten virtual maupun pada Gemini. Evaluasi dilaksanakan secara terpisah menggunakan 4 dokumen yang telah disediakan, di mana setiap dokumen akan diuji dengan 10 pertanyaan dalam 1 sesi, kemudian 10 pertanyaan yang sama akan diajukan kembali pada sesi yang berbeda. Dengan demikian, akan diperoleh 40 pertanyaan berbeda dengan total 80 pertanyaan yang diajukan. Seluruh pertanyaan disampaikan menggunakan mode *voice* pada kedua sistem, dan latensi diukur menggunakan *stopwatch* yang dimulai tepat setelah pertanyaan selesai diucapkan dan dihentikan tepat setelah respons suara diberikan. Model Gemini yang akan digunakan adalah Gemini Fast. Jawaban yang benar akan diberikan 2 poin, jawaban tidak lengkap akan diberikan 1 poin dan jawaban tidak benar diberikan 0 poin.

Setelah semua poin terkumpul, dilakukan penjumlahan semua poin dengan poin maksimal adalah 160 poin. Kemudian dilakukan rata-rata untuk mendapatkan persentase akurasi jawaban antara asisten virtual dengan Gemini.

$$Akurasi\ jawaban = \frac{\Sigma Poin}{2 \times 80} \times 100\% \quad (3-1)$$

Begitu juga dengan Latensi, setelah terkumpul maka akan dilakukan penjumlahan semua latensi dan dilakukan rata-rata dari latensi Asisten Virtual dengan Gemini.

$$Latensi = \frac{\Sigma Latensi}{80} \quad (3-2)$$



UNIVERSITAS
MA CHUNG

BAB IV

Hasil dan Pembahasan

4.1 Gambaran Umum dan Alur Web Application Asisten Virtual

Web *application* akan dibagi menjadi 2 jenis *user*, yang pertama adalah *user admin* yang memiliki akses ke semua fitur untuk mengelola pengetahuan dari asisten virtual dan dapat melakukan analitik rangkuman pertanyaan yang ditanyakan oleh mahasiswa. Pada fitur manajemen dokumen terletak pada menu *Knowledge Base*. Menu ini berisikan *list* dokumen apa saja yang sedang digunakan oleh asisten virtual. Admin dapat menambah, menghapus dan mengunduh dokumen. Untuk menerapkan jika admin melakukan perubahan dokumen pengetahuan, maka admin harus menekan tombol *Activate Documents*, maka dokumen terbaru akan diterapkan pada pengetahuan asisten virtual. Fitur ini adalah fitur yang sangat penting untuk memastikan pengetahuan asisten virtual dapat terpersonalisasi, terkelola dan terpantau dengan baik.

Fitur berikutnya yang menjadi pembeda virtual asisten pada penelitian ini dengan virtual asisten pada umumnya. Tujuan utama pengembangan asisten virtual ini adalah dapat mengelola seluruh sistem dan seluruh data secara bebas. Oleh karena itu salah satu penerapannya adalah dengan semua pertanyaan dari mahasiswa akan di rangkum dan disajikan dalam bentuk rangkuman analitik pada menu *dashboard* admin. Hasil analitik pertanyaan ini dibagi menjadi 2, yaitu pertanyaan yang dapat dijawab oleh asisten virtual yang jawabannya berasal dari *knowledge base*, dan pertanyaan yang tidak dapat dijawab karena pengetahuannya belum disediakan. Selain itu juga disajikan berapa kali pertanyaan pada kategori tertentu ditanyakan. Hal ini dapat membantu admin untuk menganalisa informasi apa saja yang dibutuhkan oleh mahasiswa.

Fitur terakhir yang hanya tersedia pada *user admin* adalah penyajian data total percakapan yang sudah masuk dan tersimpan pada vektor database. Hal ini ditujukan untuk menjadi salah satu penjagaan kualitas jawaban dari asisten virtual. Mengingat kemampuan asisten virtual pada saat ini termasuk asisten virtual pada umumnya, masih menghadapi kendala kehilangan konteks atau juga disebut dengan istilah *exhausted* atau kelelahan yang menyebabkan jawaban asisten virtual bisa keluar dari konteks. Hal ini disebabkan karena riwayat percakapan yang sudah

terlalu panjang. Meskipun saat ini untuk *user guest* atau diperuntukan untuk mahasiswa memiliki fitur *reset session* yang membuat setiap sesi baru maka akan menggunakan riwayat percakapan baru, tetapi perlu untuk sudah menyediakan fitur pembersihan data pada vektor database.

Selanjutnya terdapat *user guest* untuk mahasiswa yang hanya dapat mengakses asisten virtual untuk mengajukan pertanyaan melalui *chat* atau *voice* yang juga tersedia pada *user admin*. Pada menu Asisten Virtual ini, *user* dapat memilih mengajukan pertanyaan melalui *chat* dengan fitur STT atau melalui *voice* yang memanfaatkan STT dan TTS. Asisten virtual sudah di program untuk menjawab pertanyaan mengenai Universitas Ma Chung saja dengan sumber informasi dari *knowledge base* yang sudah dikelola oleh admin. Untuk mengakses *user guest* ini juga memerlukan *login*, jadi tidak bisa langsung diakses. Tujuannya adalah supaya asisten virtual yang ditujukan untuk Universitas Ma Chung hanya tersedia untuk layanan di Universitas Ma Chung, bukan untuk umum.

4.2 Tuning Knowledge Base

Dari hasil jawaban virtual asisten pada tahap penelitian, dengan hanya memasukkan dokumen *Student Guide* secara mentah menyebabkan proses *chunking* informasi untuk disimpan pada vektor *database* sangat bias. Hasilnya ketika dilakukan pencarian jawaban, hasilnya informasi bisa ada bagian yang hilang atau bahkan tidak ditemukan. Oleh karena itu dilakukan proses *tunning* dokumen dari yang awalnya berbentuk paragraf menjadi bentuk poin-poin sederhana. Hal ini membantu pada proses *chunking* informasi dari dokumen ke vektor *database* dari Supabase karena proses pemotongan informasi menjadi poin-poin dapat lebih terkendali dari pada dengan menggunakan paragraf. Selain itu dalam proses memasukkan pengetahuan ke vektor database Supabase itu juga memerlukan bantuan LLM untuk memastikan konteksnya tidak hilang. Pada proses ini bisa saja dapat menghabiskan *token* Gemini sebagai LLM yang cukup besar.

Bab I - PENDAHULUAN

Sub-bab 1.1 - Landasan Yuridis

1. Undang-undang Republik Indonesia Nomor 20 tahun 2003 tentang Sistem Pendidikan Nasional.
2. Undang-undang Republik Indonesia Nomor 12 tahun 2012 tentang Pendidikan Tinggi.
3. Peraturan Presiden Republik Indonesia Nomor 8 tahun 2012 tentang Kerangka Kualifikasi Nasional Indonesia (KKNI).
4. Peraturan Pemerintah Republik Indonesia Nomor 4 tahun 2014 tentang Penyelenggaraan Pendidikan Tinggi dan Pengelolaan Perguruan Tinggi.
5. Peraturan Menteri Pendidikan, Kebudayaan, Riset, dan Teknologi Republik Indonesia Nomor 41 tahun 2021 tentang Rekognisi Pembelajaran Lampau.
6. Peraturan Menteri Pendidikan, Kebudayaan, Riset, dan Teknologi Republik Indonesia Nomor 53 tahun 2023 tentang Penjaminan Mutu Pendidikan Tinggi.
7. Keputusan Direktorat Jendral Pendidikan Tinggi, Riset, dan Teknologi Nomor 210/M/2023 tentang Indikator Kinerja Utama PT dan Lembaga Layanan Pendidikan Tinggi.

Gambar 4.1 Tuning Dokumen *Knowlegde Base*

4.3 Persiapan Penelitian

4.3.1 Persiapan Perangkat

Perangkat yang digunakan pada penelitian ini adalah komputer dengan spesifikasi CPU Intel Core i7-3820, NVIDIA GeForce RTX 4060 (8GB VRAM), RAM 32 GB, HDD dengan kapasitas kosong lebih dari 80 GB, *Microphone* standar dan *Speaker* standar. Dari semua komponen tersebut yang menjadi titik pentingnya adalah pada GPU. GPU ini adalah kebutuhan mutlak untuk fitur TTS, karena tanpa adanya GPU berdasarkan percobaan yang dilakukan dengan menggunakan CPU saja, menyebabkan TTS tidak dapat digunakan. Selain itu semakin baik spesifikasi GPU-nya maka performa kecepatan TTS untuk menghasilkan suara akan semakin tinggi dan hal ini akan mempengaruhi *response time* dari asisten virtual.

Kemudian kebutuhan ruangan penyimpanan yang cukup tinggi baik di *disk system* dan *disk data*. Hal ini dikarenakan ada beberapa *software* seperti docker yang berfungsi untuk menjalankan n8n hanya dapat berjalan di *disk system* dan berdasarkan hasil penelitian ini, perkiraan total ruang yang dibutuhkan untuk docker adalah 35GB. Kebutuhan ruang penyimpanan untuk TTS juga cukup besar, selain karena *library* yang digunakan, pada penelitian ini juga dilakukan *training* melalui fitur *fine-tune* yang disediakan oleh F5-TTS akan memakan ruang penyimpanan yang sangat besar dengan perkiraan bisa melebihi 40GB tergantung pada banyaknya iterasi *checkpoint* model yang diinginkan.

4.3.2 Persiapan Software

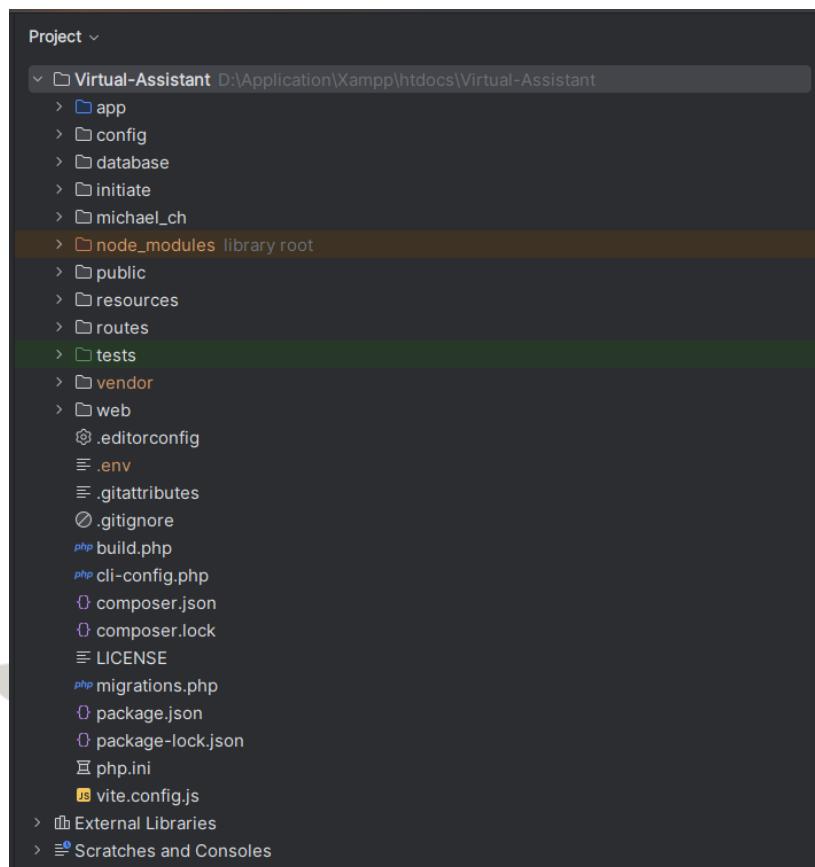
Setelah perangkat siap, maka ada beberapa software yang perlu disiapkan. Pertama adalah Integrated Development Environment (IDE) yang akan digunakan untuk mendevelop asisten virtual ada 2, yaitu PHP Storm untuk membuat web application PHP MVC dan Visual Studio Code untuk mengembangkan dan menjalankan F5_TTS. Lalu yang kedua adalah MySQL untuk database manajemennya. Kredensial dari MySQL yang dibuat pada tahap instalasi harus disimpan karena akan digunakan untuk koneksi dengan web *application*.

Untuk dapat menjalankan PHP dibutuhkan XAMPP sebagai *engine* dari PHP. Namun pada penelitian ini tidak akan menggunakan XAMPP *Control Panel* tetapi akan memanfaatkan php.exe saja. File php.exe yang diperoleh dari XAMPP perlu didaftarkan pada *system environment variables* dari windows agar PHP dapat dijalankan dengan *setting* untuk MVC. Kemudian untuk dapat melakukan instalasi dan memanajemen *library* dari PHP MVC yang akan digunakan, dibutuhkan composer. Composer ini akan di *install* pada sistem dan pada proses instalasi Composer akan membutuhkan *path* yang menuju ke php.exe sebelumnya, sehingga php.exe tersebut harus sudah ada terlebih dahulu. Kebutuhan lain untuk web *application* adalah nodeJs yang akan berfungsi untuk melakukan *compile* CSS dan JS dari web *application*.

4.3.3 Persiapan Web Application dengan PHP MVC

Untuk persiapan membuat web *application*, dilakukan *cloning repository* pribadi dari *github* untuk memperoleh kerangka kosong dari PHP MVC. Karena titik fokus penelitian ini bukan pada PHP MVC, maka akan langsung berfokus mengimplementasikan asisten virtual pada kerangka kosong untuk dapat berjalan di atas sistem PHP MVC. Kerangka kosong ini mencakup sistem *router* untuk mengatur alur masuknya *request*, pemisahan *Model*, *View* dan *Controller*, sistem *repository* untuk menangani koneksi dengan database MySQL, sistem enkripsi, sistem *middleware* dan *compiler resource* CSS dan JS. Tahap pertama yang harus dilakukan setelah melakukan *cloning* kerangka adalah mengunduh semua *library* yang dibutuhkan menggunakan Composer dengan perintah “*composer install*”. Setelah selesai maka akan terbentuk folder baru dengan nama vendor yang nantinya akan berisikan kumpulan *library* yang dibutuhkan secara otomatis. Tahap terakhir

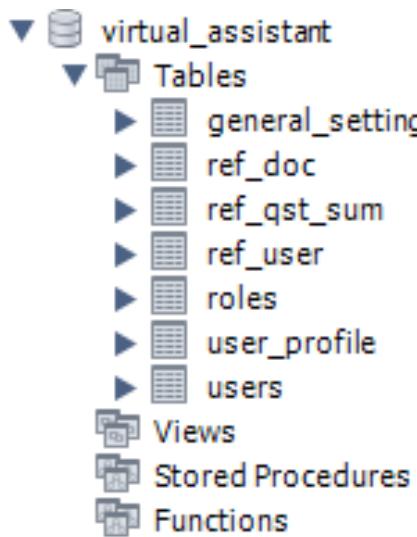
untuk persiapan PHP MVC adalah dengan mengunduh library dari NodeJS untuk dapat melakukan *compiler* CSS dan JS dengan perintah “*npm install*” dan nanti akan terbentuk folder *node_modules*.



Gambar 4.2 Struktur Framework PHP MVC

4.3.4 Persiapan Database MySQL

Untuk menyimpan data pada penelitian ini menggunakan MySQL sebagai database. Dengan menggunakan MySQL Workbench, pada penelitian ini membuat 1 skema database dengan nama *virtual_assistant*. Untuk spesifikasi tabel terdapat *general_setting* untuk menyimpan *settingan* umum dari aplikasi, *ref_doc* untuk menyimpan detail dokumen pengetahuan dari asisten virtual, *ref_qst_sum* untuk menyimpan rangkuman pertanyaan untuk analitik, *users* untuk menyimpan data *user* untuk dapat masuk ke dalam aplikasi, *roles* untuk menyimpan *list role* yang tersedia, *ref_user* sebagai penghubung many-to-many dari *users* dan *roles*, *user_profile* untuk menyimpan detail data pengguna dari *users*.



Gambar 4.3 Tabel Database Web Application

Untuk menghubungkan web *application* dengan database, maka perlu dilakukan *settingan* koneksi pada file .env pada PHP MVC. Untuk setting yang perlu dimasukan adalah DB_DRIVER yang harus diisi pdo_mysql, DB_HOST yang diisi dengan alamat *host* pada penelitian ini yaitu 127.0.0.1, kemudian DB_PORT yang diisi dengan port yang digunakan pada penelitian ini yaitu 3306, DB_DATABASE yang diisi dengan nama skema yang sudah dibuat yaitu virtual_assistant, DB_USERNAME dan DB_PASSWORD yang akan diisi dengan kredensial dari MySQL. Karena sistem PHP MVC yang saat ini digunakan mendukung pemisahan *environment* produksi dan pengembangan maka dilakukan *settingan* yang sama dengan variabel yang diakhiri dengan _DEV. Setelah semua sudah diisi dengan tepat maka sistem bawaan PHP MVC akan secara otomatis membuat koneksi dengan database.

```

DB_DRIVER="pdo_mysql"
DB_HOST="127.0.0.1"
DB_PORT="3306"
DB_DATABASE="virtual_assistant"
DB_USERNAME= [REDACTED]
DB_PASSWORD= [REDACTED]

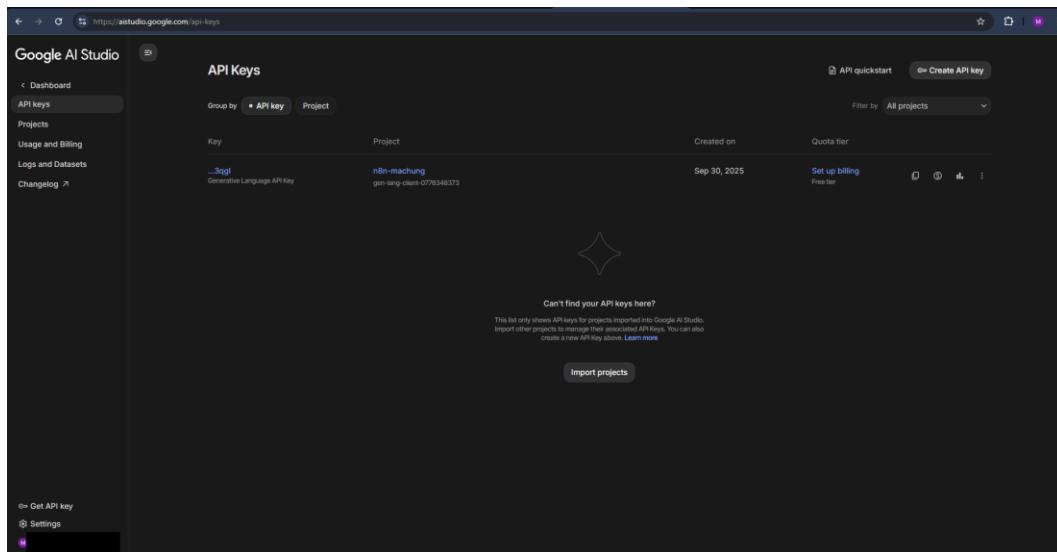
DB_DRIVER_DEV="pdo_mysql"
DB_HOST_DEV="127.0.0.1"
DB_PORT_DEV="3306"
DB_DATABASE_DEV="virtual_assistant"
DB_USERNAME_DEV= [REDACTED]
DB_PASSWORD_DEV= [REDACTED]

```

Gambar 4.4 Konfigurasi Koneksi Database

4.3.5 Persiapan API Gemini

API Gemini adalah fitur utama untuk digunakan sebagai LLM dari asisten virtual. Untuk mendapatkan API maka dilakukan pendaftaran pada Google AI Studio. Setelah berhasil mendaftar maka akan terbentuk 1 API Key yang akan digunakan sebagai LLM dari asisten virtual di n8n nanti.

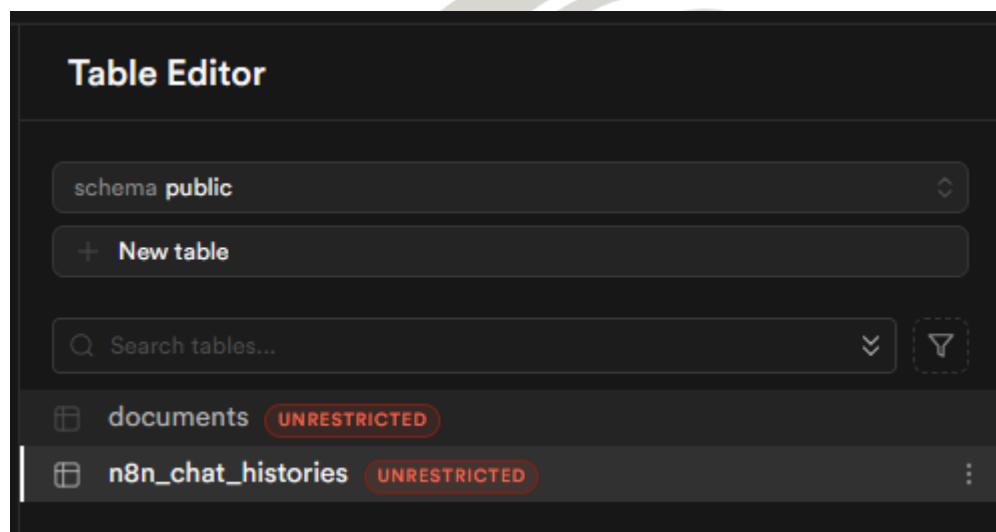


Gambar 4.5 Google API Studio

4.3.6 Persiapan Vektor Database Supabase

Supabase yang berperan sebagai vektor database untuk menyimpan pengetahuan dan sebagai Postgres database untuk menyimpan data riwayat

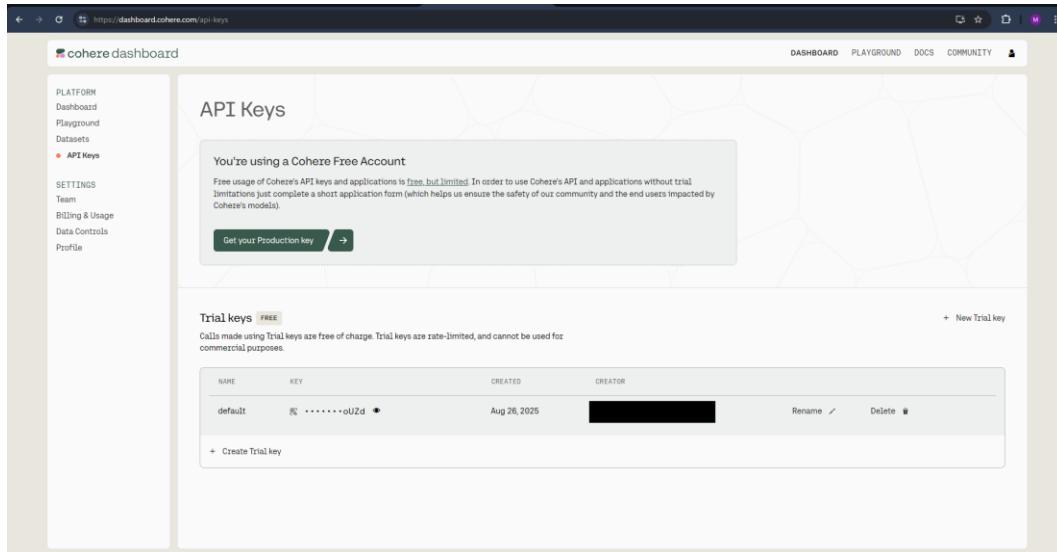
percakapan yang perlu dipersiapkan terlebih dahulu. Melalui *website* resmi dari Supabase dilakukan pendaftaran dahulu. Setelah terdaftar dan dapat membuat projek maka dilakukan pembuatan tabel *documents* untuk menyimpan pengetahuan asisten virtual dan tabel *n8n_chat_history* untuk menyimpan riwayat percakapan. Tabel *documents* memiliki spesifikasi wajib berupa *ide*, *content*, *metadata* dan *embedding*. Tabel *n8n_chat_history* memiliki spesifikasi *ide*, *session_id* dan *message*. Setelah semua tabel siap, maka dilakukan pembuatan *secret key* untuk menghubungkan antara *n8n* dengan Supabase. Untuk riwayat percakapan akan disimpan menggunakan koneksi dari postgres yang bisa dilihat pada *setting project* dari Supabase.



Gambar 4.6 Tabel Vektor Database Supabase

4.3.7 Persiapan Reranker

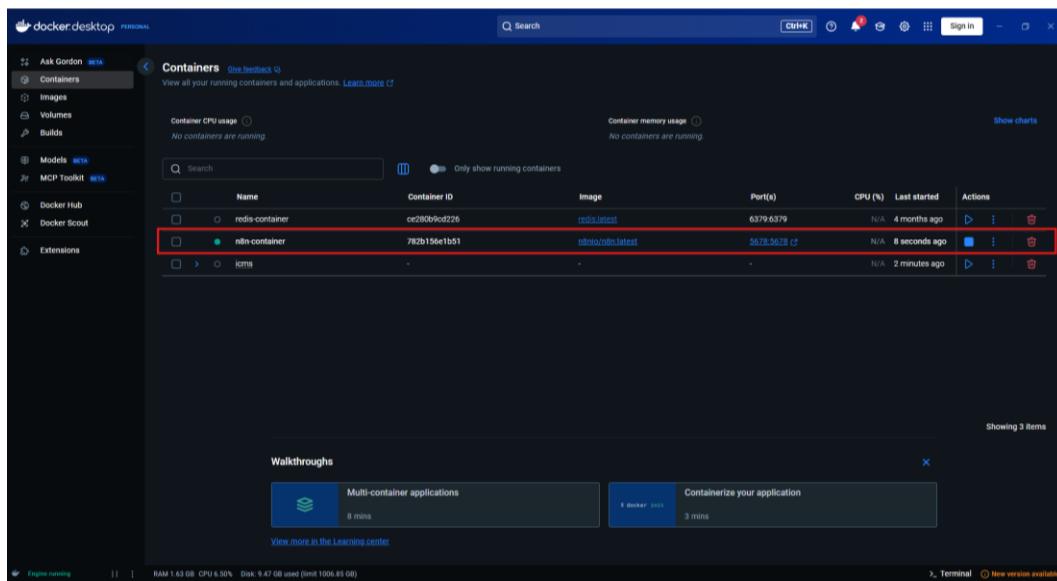
Reranker yang berfungsi untuk memberikan peringkat tingkat relevan pencarian jawaban untuk meningkatkan keakuratan jawaban dari asisten virtual. Untuk dapat menggunakan Reranker dilakukan pendaftaran pada *website* resmi dari Cohere. Setelah terdaftar, dilakukan pembuatan API key yang akan digunakan untuk menghubungkan *n8n* dengan Reranker.



Gambar 4.7 API Cohere Reranker

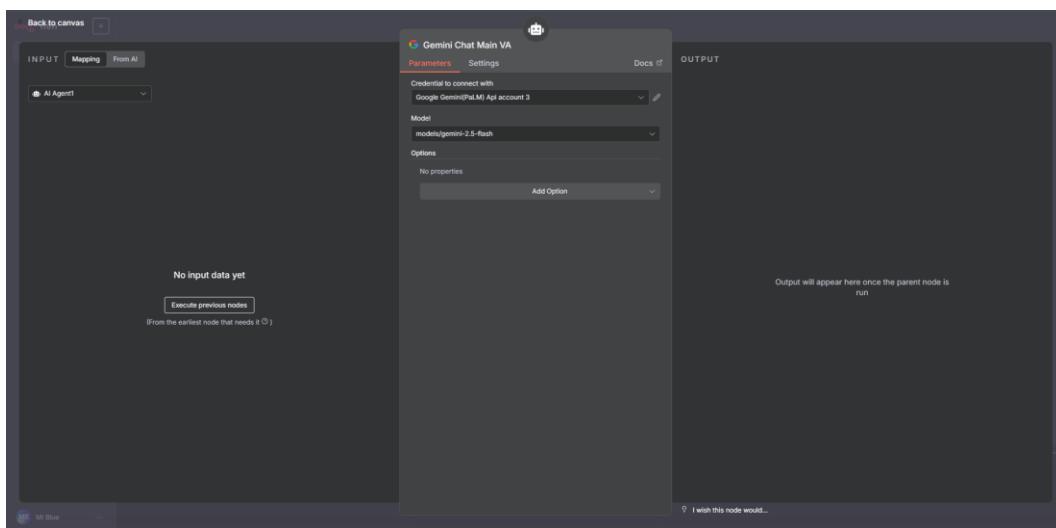
4.3.8 Persiapan N8N

N8N adalah inti utama dari kecerdasan asisten virtual yang akan memanajemen semua sistemnya mulai dari *input*, pengolahan LLM, pencarian informasi pada Supabase dan pengembalian *response*. Untuk menjalankan n8n perlu menggunakan Docker. Saat Docker sudah berjalan, dilakukan *pull image* dari n8n dengan nama *image* n8nio/n8n. Setelah *image* selesai di *pull*, maka dilakukan pembuatan *container* dengan *settingan port* n8n nantinya akan berjalan pada *port* 5678. Setelah *container* terbentuk, maka n8n sudah bisa dijalankan diatas Docker *engine*.

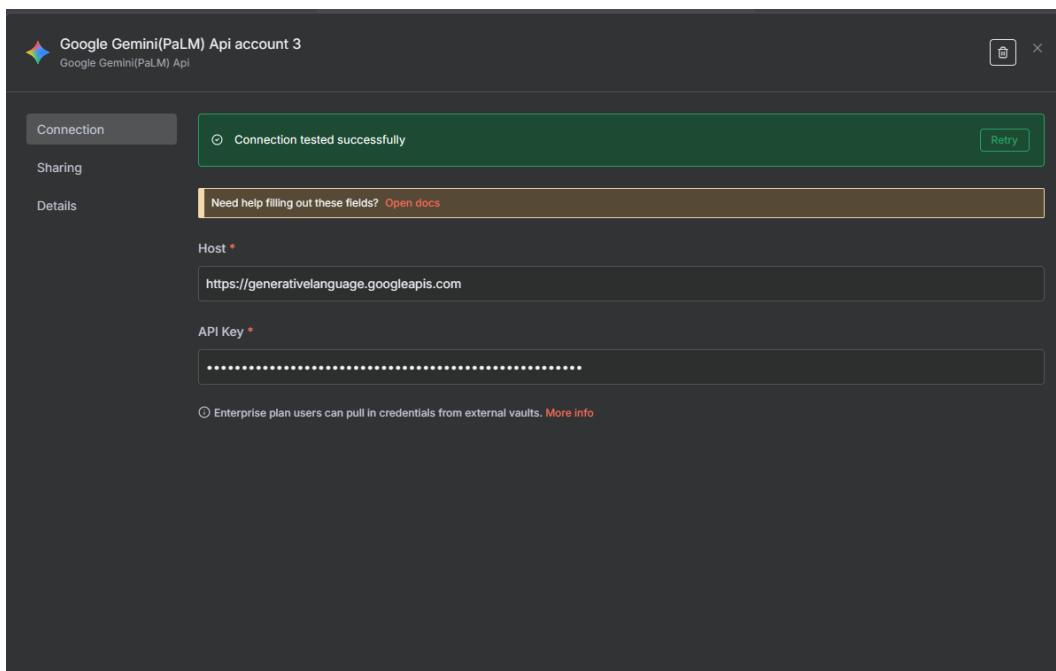


Gambar 4.8 Docker Container n8n

Setelah n8n sudah berjalan diatas Docker, selanjutnya adalah dengan melakukan pengaturan untuk *node* yang mengarah ke *service*. *Service* eksternal pertama adalah *node* untuk Gemini yang berperan sebagai LLM. Pada *node* Gemini dilakukan “*Create new credential*” dan pada *field API Key* akan diisi dengan API yang diperoleh pada Google API Studio sebelumnya. Setelah terhubung maka dilakukan pengaturan untuk model Gemini yang akan digunakan, dan pada penelitian ini akan menggunakan Gemini-2.5-flash.

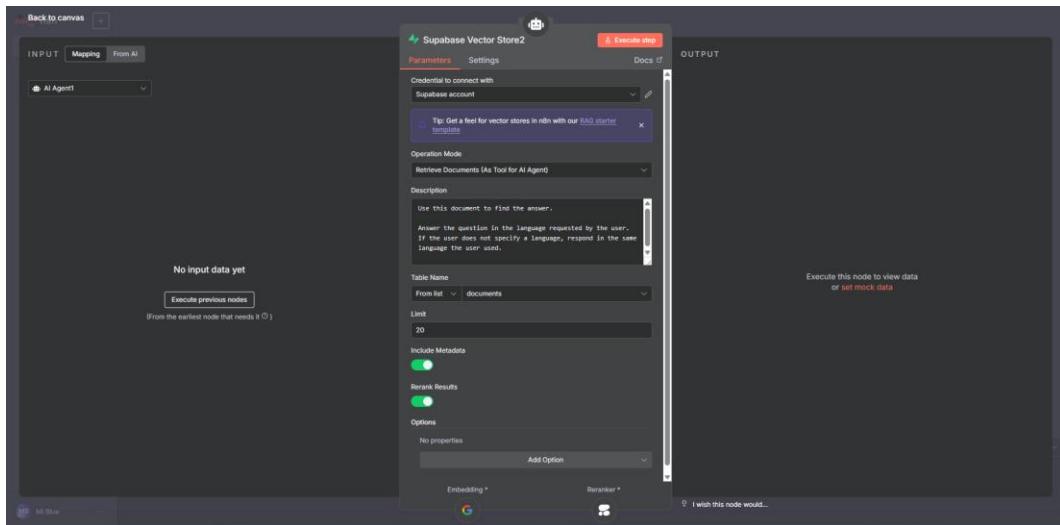


Gambar 4.9 Setting Google Gemini Node

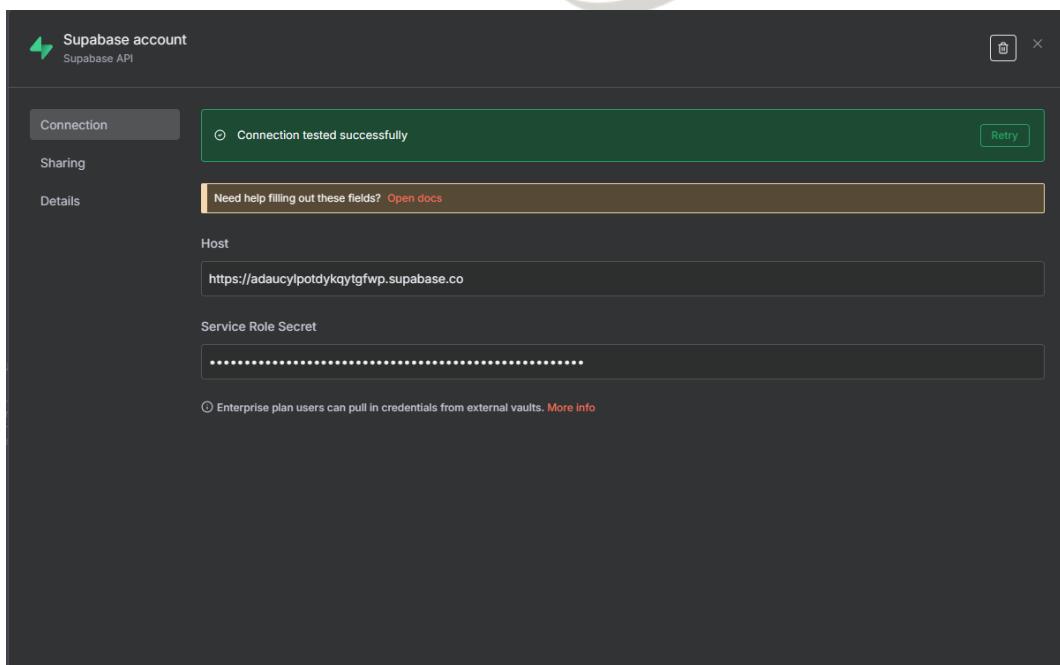


Gambar 4.10 Setting Kredensial Gemini

Selanjutnya yang juga penting adalah pengaturan untuk *node* Supabase *Vektor Store* untuk vektor database. Dari kredensial yang diperoleh dari website Supabase resmi di masukan ke dalam pengaturan koneksi node Supabase. Setelah berhasil terhubung, dilakukan pengaturan nama tabel yang menjadi tujuan untuk menyimpan potongan informasi pengetahuan, dalam hal ini nama tabelnya adalah tabel *document*.



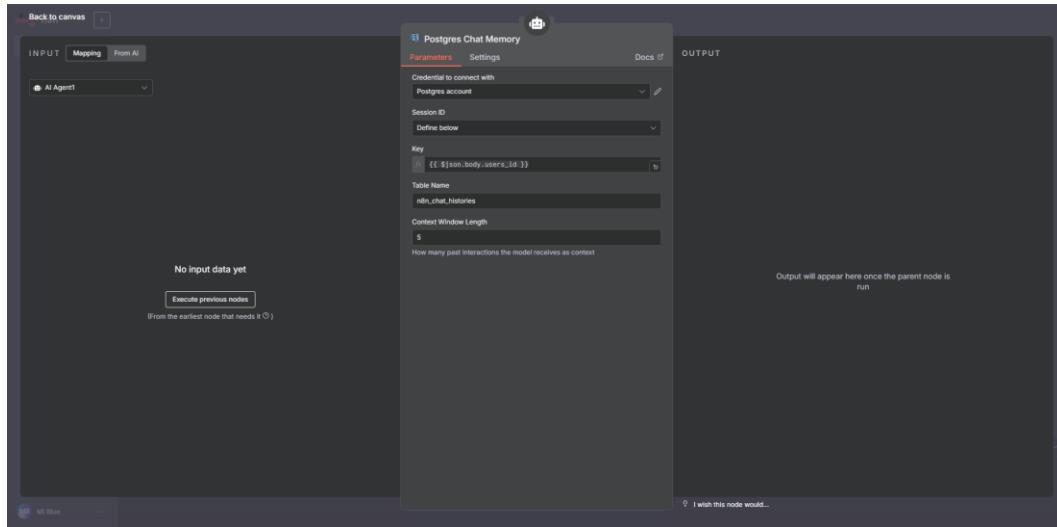
Gambar 4.11 Setting Node Supabase



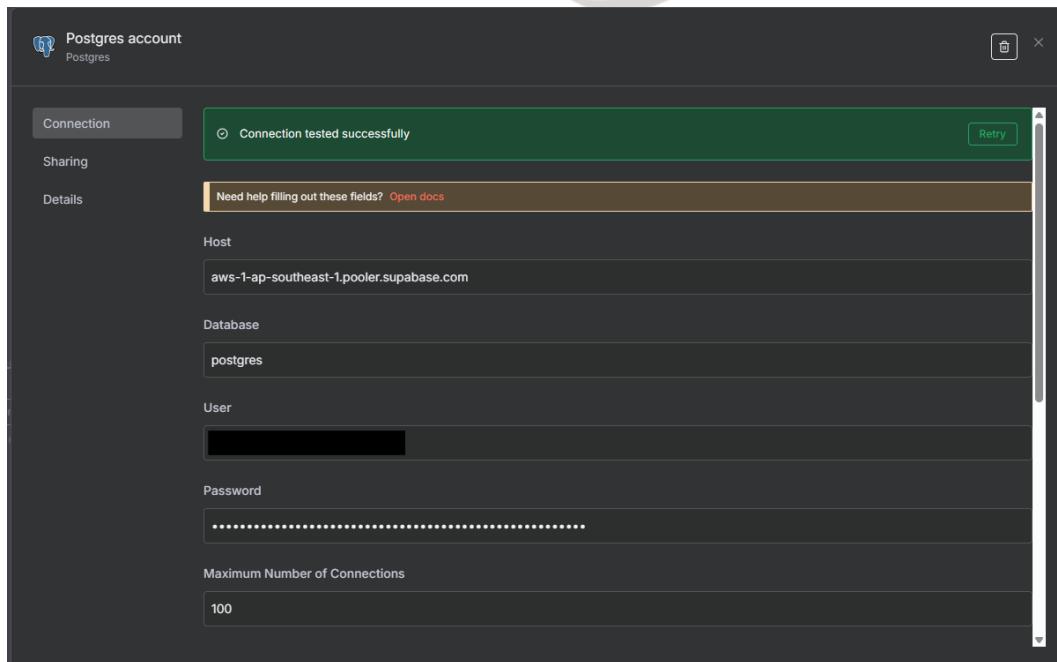
Gambar 4.12 Setting Kredensial Supabase

Untuk menyimpan riwayat percakapan juga tetap membutuhkan vektor database Supabase namun dengan pengaturan yang berbeda. Untuk dapat terhubung

memerlukan node Postgres Chat Memory. Cara untuk terhubung dengan Supabase juga berbeda dengan node Supabase Vektor Store. Untuk mendapatkan kredensial koneksi bisa dengan mengakses pengaturan koneksi database dari Supabase. Dengan menggunakan database Postgres dan method session pooler untuk digunakan sebagai pengaturan kredensial node Postgres Chat Memory.

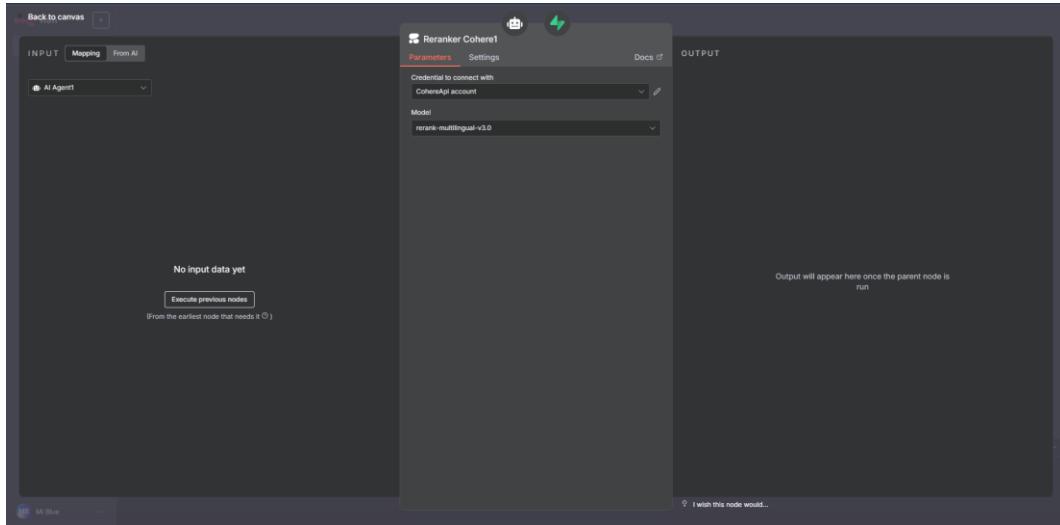


Gambar 4.13 Setting Node Postgres *Chat Memory*

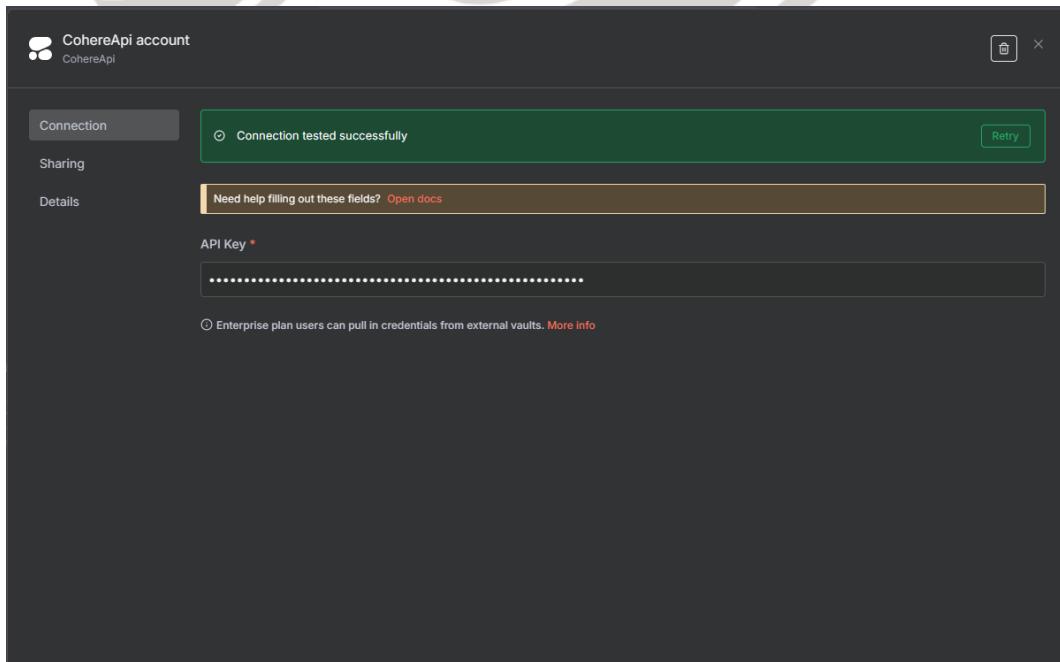


Gambar 4.14 Setting Kredensial Node Postgres *Chat Memory*

Terakhir adalah pengaturan untuk Reranker yang akan digunakan untuk memberikan peringkat jawaban asisten virtual yang didapat dari vektor database Supabase. Dengan menggunakan API yang didapat dari halaman resmi Cohere Reranker, API Key tersebut akan digunakan untuk mengisi *field API Key*.



Gambar 4.15 *Setting Node Reranker*



Gambar 4.16 Setting Kredensial Reranker

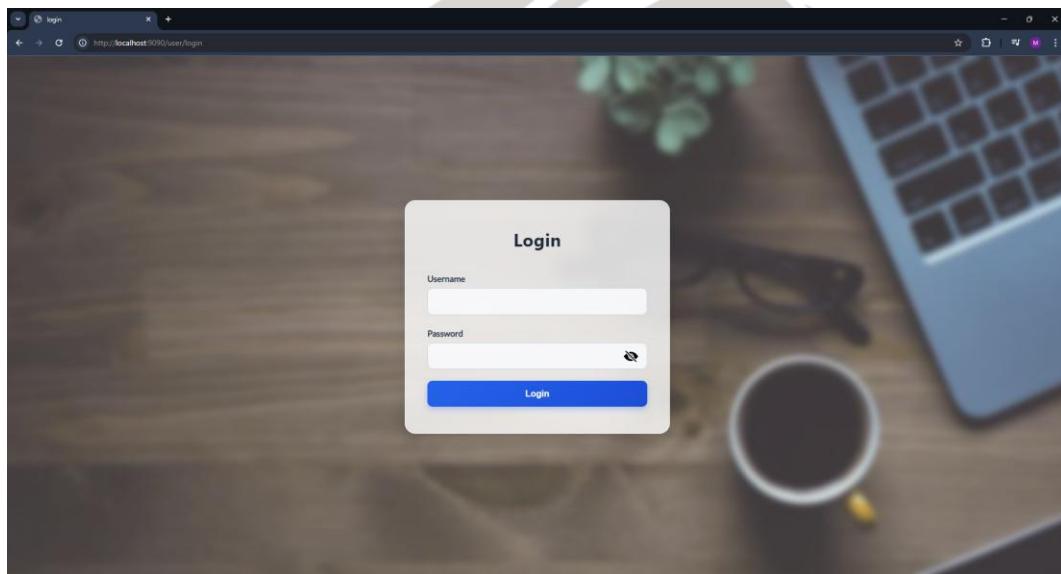
4.3.9 Persiapan F5_TTS

F5_TTS adalah fitur TTS dari asisten virtual. Untuk menjalankannya diperlukan python versi 3.10.11 yang menjadi kebutuhan dari F5_TTS dan CUDA GPU driver untuk F5_TTS dapat terhubung dan menggunakan GPU dalam proses

penghasilan suara. Setelah semua sudah siap maka dengan menggunakan IDE Visual Studio Code langkah pertama yang harus disiapkan untuk menjalankan F5-TTS adalah virtual environment dari python 3.10.11. Kegunaan virtual environment ini untuk mengisolasi aplikasi dan library yang akan dijalankan oleh F5_TTS. Untuk virtual environment yang digunakan pada penelitian ini adalah VENV dari python 3.10.11. Setelah VENV terpasang maka dilakukan clone repository dari Github resmi F5_TTS dengan author atau pemilik utama bernama SWivid.

4.4 Hasil Web Application

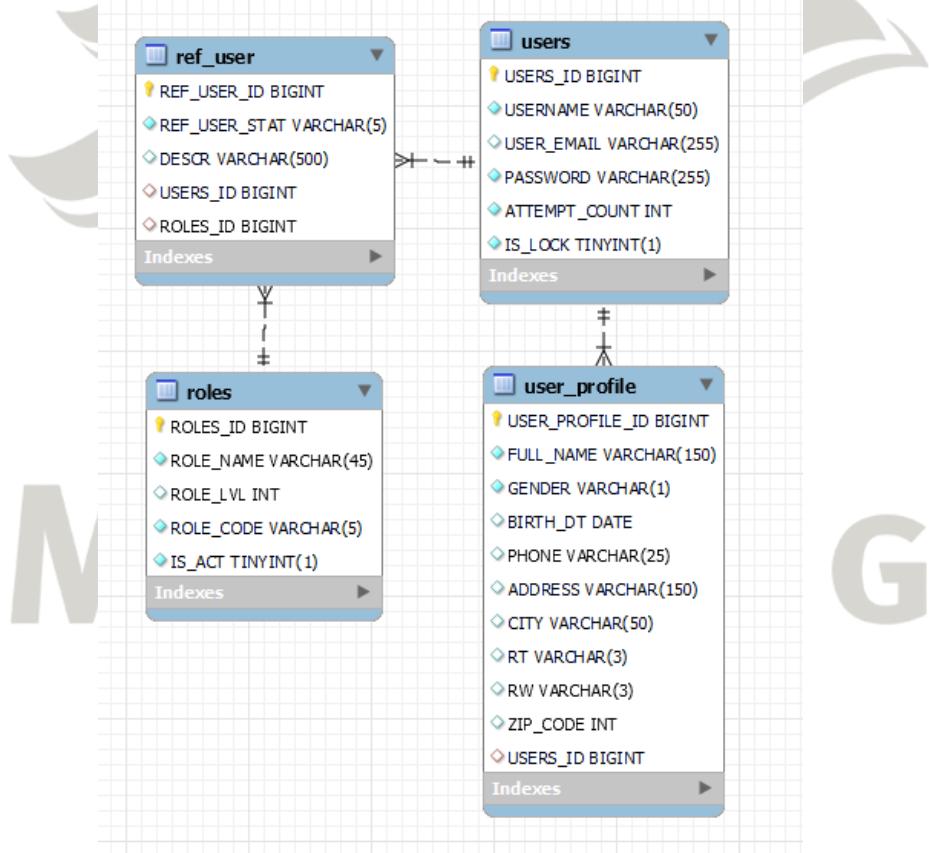
4.4.1 Halaman Login



Gambar 4.17 Halaman Login Web Application

Untuk dapat mengakses asisten virtual, *user* wajib melakukan *login* baik admin maupun *guest* atau dalam konteks penelitian ini adalah mahasiswa. Untuk versi saat ini tidak terdapat fitur registrasi atau penambahan *user* dan manajemen *user role*, karena tujuan penelitian ini adalah mengembangkan fondasi utama dari asisten virtual yang dapat dipersonalisasi. Karena tidak ada fitur penambahan user maka secara *default* ketika *web application* di jalankan akan memiliki 2 *user* bawaan yaitu *user admin* dan *user guest*. Cara pemisahan *user* dengan menyimpan data *user* pada database dan *user* akan memiliki *role* untuk memisahkan *admin* dan *guest*.

Untuk spesifikasinya terdapat tabel users yang akan menyimpan data untuk *login* dengan kolom USERS_ID, USERNAME, USER_EMAIL, PASSWORD, ATTEMPT_COUNT, IS_LOCK. Kemudian terdapat tabel roles yang akan menyimpan data role dengan kolom ROLES_ID, ROLE_NAME, ROLE_LVL, ROLE_CODE, IS_ACT. Karena hubungan antara users dan roles adalah *many-to-many* maka diperlukan tabel ref_user sebagai tabel *mapping* antara tabel users dan roles dengan kolom REF_USER_ID, REF_USER_STAT, DESCR, USERS_ID, ROLES_ID. Terdapat 1 tabel lagi yang untuk saat ini belum digunakan namun disiapkan untuk kebutuhan dimasa depan untuk menyimpan data pengguna dari *user web application*, tabel tersebut adalah tabel user_profile dengan kolom USER_PROFILE_ID, FULL_NAME, GENDER, BIRTH_DT, PHONE, ADDRESS, CITY, RT, RW, ZIP_CODE, USERS_ID.



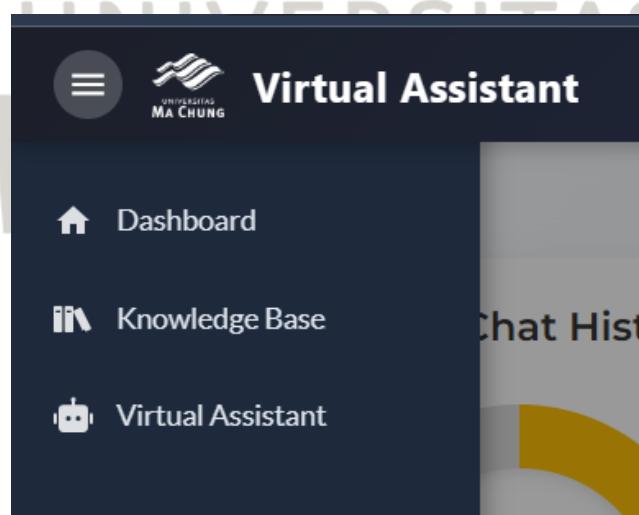
Gambar 4.18 Spesifikasi Database *User Management*

Untuk keamanan proses *login*, digunakan enkripsi *password* menggunakan algoritma AES-256 dengan *key* enkripsi yang akan disimpan pada .env dari PHP MVC. Untuk memanajemen sesi *user* agar *user guest* menggunakan fungsi

`$_SESSION` dari PHP untuk menyimpan sesi *user* setelah *login*. Kemudian untuk menjamin keamanan halaman admin supaya tidak dapat diakses oleh *user guest* menggunakan *middleware* yang akan menjaga *request* masuk apabila sesi bukan milik admin maka akan langsung ditolak. Jika terdapat pengguna yang memaksa *login* juga terdapat fitur *locking* yang akan otomatis mengunci *user* yang dilakukan tindakan *brute force*. Dalam penelitian ini ditetapkan sebanyak 3 kali percobaan gagal maka *user* tersebut akan otomatis terkunci dan tidak dapat dilakukan *login*. Untuk membuka kunci hanya dapat dilakukan dari *database* oleh *database administrator*.

Setelah berhasil *login*, *user* dengan *role* admin akan otomatis masuk ke halaman *dashboard* admin sedangkan *user* dengan *role* *guest* akan langsung mengarah ke halaman *virtual assistant*. Cara pemisahan ini sudah akan otomatis ditangani oleh *controller authentication* dan dengan kombinasi *middleware*. Dengan bantuan *middleware*, apabila *user* mengetahui url dari halaman admin dan mencoba mengaksesnya maka akan otomatis dialihkan kehalaman *exception 404*. Halaman dibuat eror 404 untuk menjaga alamat halaman admin tidak terlihat secara langsung yang akan mempersusah jika ada yang ingin melacak alamat halaman admin.

4.4.2 Navigasi dan Header



Gambar 4.19 Sidebar Navigasi Admin

Setelah *login*, terdapat pembeda antara *user* admin dan *user guest*. Pada *user* admin terdapat fitur navigasi *sidebar* untuk melakukan navigasi ke halaman admin

lainnya. *Sidebar* ini tidak akan muncul pada sesi *user guest*. Selain terdapat juga header yang berisikan logo, nama aplikasi, nama sesi *user* dan tombol *logout* untuk admin atau *new session* untuk *guest*. Pada halaman admin, semua *history* percakapan dengan asisten virtual hanya terikat pada 1 sesi *id login* admin. Admin dapat dengan mudah melakukan *logout* dengan menekan tombol *logout*. Hal ini berbeda dengan *user guest* yang penggunanya bisa banyak orang, sehingga apabila berganti pengguna perlu dilakukan reset sesi dengan tombol *new session*. Pada sesi *guest* untuk melakukan *logout* tidak langsung tersedia kasat mata. Untuk melakukan *logout*, *user* perlu menahan tombol *new session* selama kurang lebih 5 detik. Hal ini bertujuan agar pengguna umum tidak dapat melakukan *logout* sembarangan dan berakhir asisten virtual tidak dapat diakses hingga admin atau orang yang mengetahui kredensial *login* melakukan *login* kembali. Selain itu terdapat 1 fitur tidak kasat mata lagi adalah fitur *fullscreen* dengan melakukan ketuk 2 kali pada area header. Namun fitur ini kurang optimal pada penelitian kali ini yang masih menggunakan konsep *Multi Page Application* (MPA).

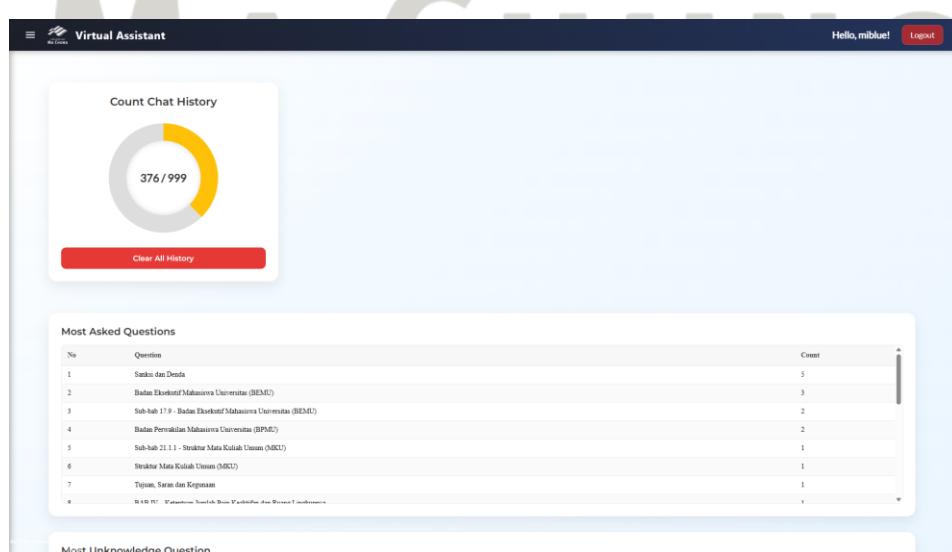


Gambar 4.20 Header Admin



Gambar 4.21 Header Guest

4.4.3 Halaman Dashboard Admin



Gambar 4.22 Halaman *Dashboard* Admin

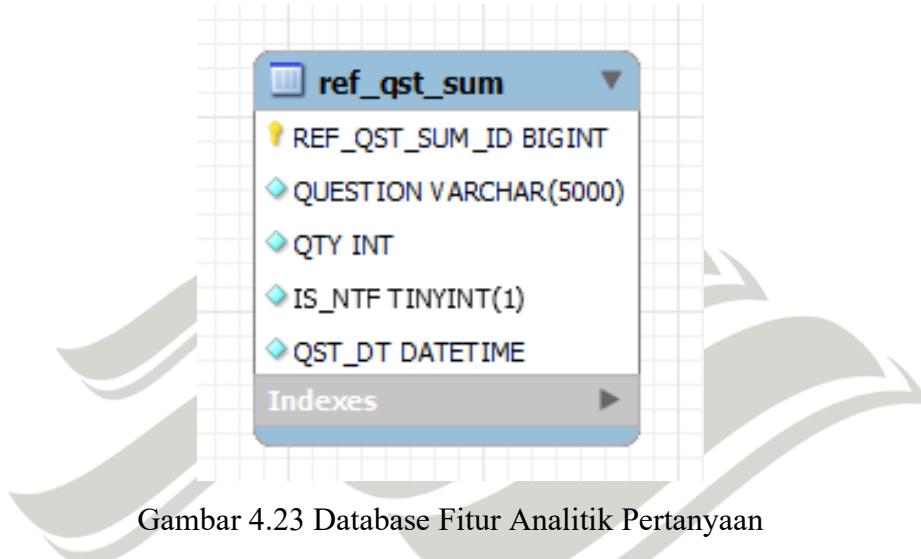
Pada halaman *dashboard* admin pada penelitian ini terdapat 2 fitur utama yang menjadi salah satu contoh bentuk penerapan personalisasi asisten virtual yang tidak dapat dilakukan pada asisten virtual umum lainnya. Fitur yang pertama adalah analitik jumlah riwayat percakapan asisten virtual dengan *user*. Tujuan dari fitur ini adalah untuk mengontrol asisten virtual agar tidak terjadi *exhausted* atau keadaan dimana asisten virtual sudah kehilangan konteks. Selain itu dengan adanya fitur ini maka admin dapat mengetahui seberapa tinggi pertanyaan yang diajukan oleh mahasiswa. Semakin tinggi jumlah riwayat pertanyaannya bisa menjadi salah satu tolak ukur apabila mahasiswa sering bertanya atau membutuhkan layanan informasi dari asisten virtual.

Untuk membuat fitur analitik jumlah riwayat percakapan tersebut langsung dengan menggunakan *webhook* pada n8n. Web application PHP akan mengirimkan *request* ke *webhook* n8n lalu n8n akan melakukan *query* ke *vector database* Supabase untuk mendapatkan semua *row* percakapan yang tersimpan dan menghitung semua *row* yang tersimpan. Setelah dihitung, hasil numeriknya dikembalikan ke web *application* PHP. Setelah PHP menerima *response* baru hasil numerik tersebut ditampilkan pada halaman menggunakan bantuan javascript.

Fitur berikutnya adalah rangkuman jenis pertanyaan yang diajukan oleh mahasiswa. Selain mengetahui jumlah pertanyaan yang masuk, perlu diketahui pertanyaan seperti apa yang diajukan oleh mahasiswa. Hal ini sangat penting untuk mengetahui informasi seperti apa yang dibutuhkan mahasiswa. Pada fitur ini admin dapat melihat jenis pertanyaan apa saja dan berapa banyak pertanyaan dengan jenis tersebut ditanyakan. Fitur analitik ini juga dibagi menjadi 2, jenis pertanyaan yang dapat dijawab oleh asisten virtual atau pertanyaan yang jawabannya sudah tersedia pada dokumen *knowledge base* dan jenis pertanyaan yang tidak dapat dijawab oleh asisten virtual. Dari sini admin dapat mengetahui informasi seperti apa yang dibutuhkan oleh mahasiswa.

Karena hasil proses pemisahan kategori dan jumlah pertanyaan pada setiap kategori harus selalu disimpan maka dibutuhkan *database*. Dibentuklah satu tabel ref_qst_sum yang berfungsi untuk menyimpan rangkuman tersebut. Spesifikasi dari tabel ref_qst_sum memiliki kolom REF_QST_SUM_ID sebagai *primary key*,

QUESTION untuk menyimpan pertanyaan, QTY untuk menyimpan jumlah pertanyaan, IS_NTF sebagai *flag* apakah pertanyaan tersebut terdapat jawabannya pada *knowledge base* atau tidak dengan 1 atau true mempresentasikan tidak ada dan 0 atau false mempresentasikan ada dan QST_DT menyimpan waktu pertanyaan tersebut diajukan. Untuk saat ini QST_DT hanya sebagai kolom waktu yang dapat dimanfaatkan di pengembangan berikutnya.



Gambar 4.23 Database Fitur Analitik Pertanyaan

Untuk prosesnya, sistem akan melakukan pencarian jawaban melalui n8n berdasarkan pertanyaan pengguna. Jika jawaban pertanyaan tersebut ditemukan, maka n8n akan melakukan *request insert* data berupa judul dari sub-bab yang memiliki poin jawabannya. Jika tidak ada sub-babnya maka akan otomatis membaca judul dari babnya. Pada proses *insert* apabila sudah ditemukan data dengan kategori yang hendak di *insert* maka akan melakukan *update* kolom QTY menjadi *increment 1*. Jika tidak ditemukan baru dilakukan *insert* data baru dengan nilai awal QTY adalah 1. Jika jawaban tidak ditemukan, maka akan dilakukan *insert* dengan cara yang sama, namun akan ditambahkan *flag* IS_NTF dengan *value true* atau 1. Kondisi terakhir, jika pengguna hanya melakukan *small talk* atau bukan pertanyaan akan informasi, maka tidak dilakukan *insert* apapun. Proses pemilihan ini dilakukan otomatis dengan menggunakan LLM gemini untuk bisa mendapatkan konteks pertanyaan, jawaban dan pencarian pada dokumen di Supabase. Dengan demikian semua proses analitik tersebut sudah berhasil dilakukan otomatis oleh sistem.

4.4.4 Halaman Knowledge Base Admin

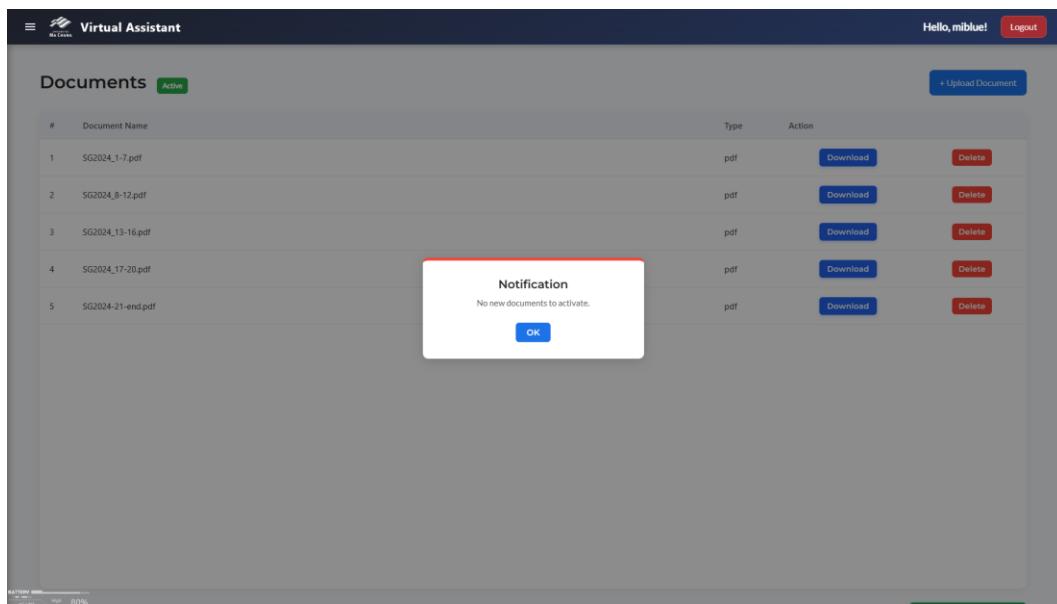
The screenshot shows a web-based application interface for managing documents. At the top, there's a header with the 'Virtual Assistant' logo, a 'Hello, miblue!' greeting, and a 'Logout' button. Below the header, the title 'Documents' is displayed with an 'Active' status indicator. A blue button labeled '+ Upload Document' is located in the top right corner. The main content area is a table listing five documents:

#	Document Name	Type	Action
1	SG2024_1-7.pdf	pdf	<button>Download</button> <button>Delete</button>
2	SG2024_8-12.pdf	pdf	<button>Download</button> <button>Delete</button>
3	SG2024_13-16.pdf	pdf	<button>Download</button> <button>Delete</button>
4	SG2024_17-20.pdf	pdf	<button>Download</button> <button>Delete</button>
5	SG2024_21-end.pdf	pdf	<button>Download</button> <button>Delete</button>

At the bottom right of the table area, there's a green button labeled 'Activate Documents'. The background of the page features a large watermark-like text 'UNIVERSITAS MA CHUNG'.

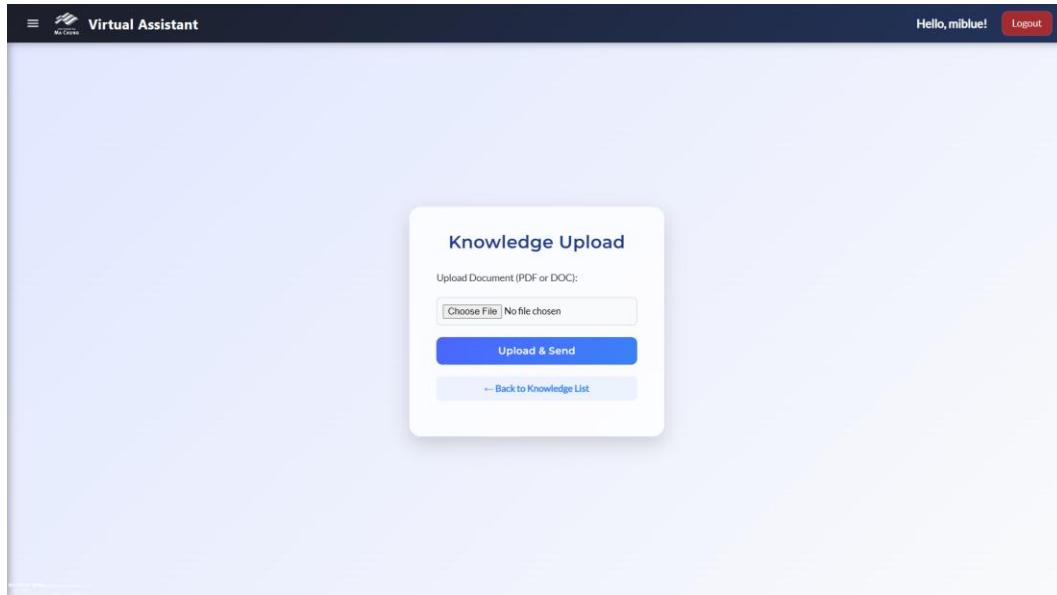
Gambar 4.24 Halaman List Dokumen Knowledge Base Admin

Pada halaman menu *Knowledge Base* yang hanya dapat diakses oleh admin, berfungsi untuk melakukan manajemen dokumen yang akan menjadi sumber pengetahuan asisten virtual. Halaman ini menampilkan *list* dokumen yang digunakan. Admin dapat mengunduh dokumen, menghapus dari *list*, menambahkan dokumen kedalam *list* dan mengaktifkan dokumen jika ada perubahan. Terdapat label *Active* yang menandakan dokumen saat ini adalah dokumen yang sedang digunakan oleh asisten virtual sebagai sumber pengetahuannya. Jika sudah aktif namun admin tetap menekan tombol *activate documents*, maka sistem akan memunculkan *popup* pemberitahuan dan tidak akan mengunggah dokumen ke n8n untuk dijadikan pengetahuan pada Supabase. Hal ini sangat penting agar sistem tidak menghabiskan *token* yang bisa saja cukup besar secara sia-sia karena memasukkan dokumen yang sama.



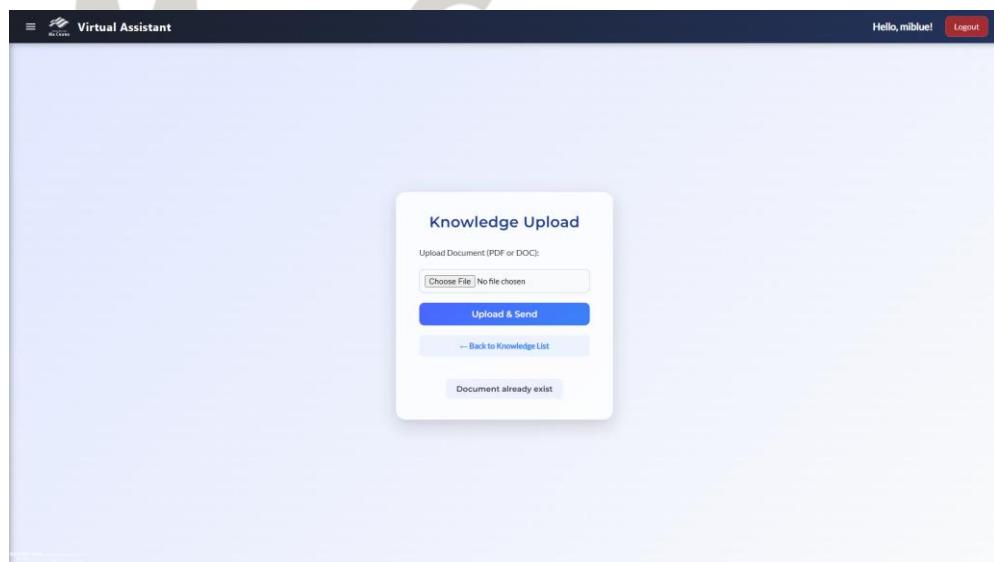
Gambar 4.25 Notifikasi Sistem Pemberitahuan

Jika admin melakukan perubahan dokumen meskipun kondisinya adalah admin menghapus dokumen dan kembali mengunggah dokumen yang sama persis, akan tetap dianggap sebagai perubahan karena sistem tidak dapat mengetahui isi dokumen tersebut bisa saja terdapat perubahan walaupun dengan nama yang sama. Karena dianggap adanya perubahan maka label *Active* akan berubah menjadi *Need Activated* sebagai penanda jika ada perubahan dokumen. Jika sistem ingin mengetahui isi dokumen, sistem perlu membaca dahulu isi dokumen dan hal ini tentunya membutuhkan bantuan Gemini sebagai LLM yang pastinya akan memakan kuota *token*. Karena keterbatasan jumlah kuota yang dimiliki pada proses penelitian ini, jadi fitur dapat membaca dokumen secara langsung tidak dilakukan untuk menghemat *token*.



Gambar 4.26 Halaman Unggah Dokumen

Berikutnya, pada halaman *list* dokumen tersebut jika tombol *Upload Document* ditekan, maka *web application* akan berpindah ke halaman unggah dokumen. Admin dapat mengunggah dokumen baru dengan tipe dokumen PDF atau DOC saja pada penelitian saat ini. Pada proses ini terdapat fitur penjagaan agar dokumen yang diunggah tidak menggunakan nama yang sama. Jika admin memasukkan dokumen dengan nama yang sama maka akan muncul pesan “*Document already exist*” dan dokumen tidak akan terunggah. Jika terjadi kegagalan dalam proses unggah atau berhasil mengunggah, maka akan selalu ditampilkan pesan pemberitahuan yang sesuai dengan kondisinya.



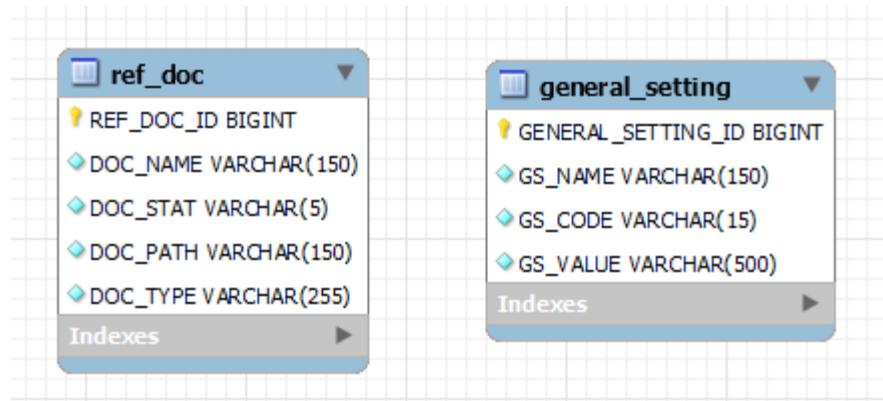
Gambar 4.27 Halaman Berhasil Unggah Dokumen

Cara memanajemen dokumen ini memerlukan bantuan database karena dokumen yang tersimpan pada vektor database Supabase sudah berupa dokumen dan tidak terdapat informasi nama dokumennya, melainkan hanya potongan-potongan informasi yang sudah diproses oleh LLM.

	content	metadata
1028	Bab I - PENDAHULUAN Sub-bab 1. Landasan Juridik 1. Undang-undang Republik Indonesia Nomor 20 tahun 2003 tentang Sistem Pendidikan Nasional. 2. Undang-undang Republik Indonesia No	{"loc": "lines": ["to": 16, "from": 1], "pdf": "in"
1029	7. Keputusan Direktorat Jenderal Pendidikan Tinggi, Riset, dan Teknologi Nomor 210/M/2023 tentang Indikator Kinerja Utama PT dan Lembaga Layanan Pendidikan Tinggi. 8. Keputusan Direktorat	{"loc": "lines": ["to": 29, "from": 14], "pdf": "in"
1030	008/MACHUNG/SK-REK/VIII/2024 tentang Penetapan Logo Baru untuk Fakultas dan Program Studi di Universitas Ma Chung. 12. Surat Keputusan Rektor Universitas Ma Chung Nomor 0051/A-U	{"loc": "lines": ["to": 43, "from": 27], "pdf": "in"
1031	Sub-bab 1.2 - Ketentuan Umum 1. Nama Universitas adalah Universitas Ma Chung. 2. Fakultas: Bagian dari Universitas yang terdiri dari beberapa Program Studi. 3. Program Studi: Keasutan kegiatan	{"loc": "lines": ["to": 60, "from": 45], "pdf": "in"
1032	9. Dosen Penasihat Akademik (Dosen PA): Dosen yang membimbing mahasiswa dalam rencana studi. 10. Tenaga Kependidikan (tenik): Karyawan non akademik. 11. Mahasiswa: Peserta didik WN	{"loc": "lines": ["to": 73, "from": 58], "pdf": "in"
1033	19. IPS (Indeks Prestasi Semester): Hasil penilaian capaian pembelajaran setiap semester. 20. IPK (Indeks Prestasi Kumulatif): Hasil penilaian capaian pembelajaran pada akhir studi. 21. Tugas Akhir	{"loc": "lines": ["to": 86, "from": 72], "pdf": "in"
1034	2. Terdiri atas tiga gelombang air (Tri Dharma Perguruan Tinggi). 3. Gelombang membentuk gunung (mewakili alam, Kota Malang, harapan jadi mewujudkan pendidikan). 4. Warna biru dan hijau (fl)	{"loc": "lines": ["to": 108, "from": 88], "pdf": "in"
1035	tengah, dengan warna dasar berbeda: a. Fakultas Ekonomi dan Bisnis Merah. b. Fakultas Teknologi dan Desain: Hijau. c. Fakultas Bahasa: Putih. d. Fakultas Ilmu Kesehatan: Kuning. Sub-bab 1.6 -	{"loc": "lines": ["to": 122, "from": 104], "pdf": "in"
1036	1. Ma Chung adalah sekolah bersejarah di Malang sejak 1950-an. 2. Ide pendirian dicetuskan pada reuni akbar HUT ke-55 SMA Ma Chung di Xianren, China pada bulan September 2001. 3. Dilandai	{"loc": "lines": ["to": 123, "from": 120], "pdf": "in"
1037	4. PT Ma Chung didirikan pada 1 Mei 2004. 5. Para pelopor adalah Soegeng Hendarto, Mochtar Riyadi, Teguh Kinarto, Hendro Sunjito, Koentjoro Loekito, Effendy Sudarmo, Agus Chandra, Hadji V	{"loc": "lines": ["to": 140, "from": 125], "pdf": "in"
1038	kebenaran. 2. Penyelenggaran didasarkan pada kedaulatan Tuhan dan berorientasi pada masyarakat. 3. Tujuan pendidikan adalah mencerdaskan dan mempersiapkan peserta didik menjadi orang	{"loc": "lines": ["to": 154, "from": 137], "pdf": "in"
1039	kewirausahaan. 3. Mendorong sikap kritis-prinsipil dan kreatif-realistik. 4. Menghasilkan lulusan siap pakai berkualitas tinggi. 5. Berperan aktif meningkatkan peradaban dunia. 6. Melaksanakan p	{"loc": "lines": ["to": 162, "from": 162], "pdf": "in"
1040	6. Ramah dan Meryenangkan. Lingkungan yang terlihat dan penuh kegembiraan. 7. Mertocatio: Menghargai prestasi, kerja keras, dan kontribusi nyata. 8. Profesional: Mengedepankan kualitas dr	{"loc": "lines": ["to": 180, "from": 164], "pdf": "in"
1041	Program Diploma III (D3). Sub-bab 1.15 - Fakultas dan Program Studi 1. Fakultas Ekonomi dan Bisnis memiliki program studi a. Program Studi SI Manajemen b. Program Studi SI Akuntansi c. Progr	{"loc": "lines": ["to": 199, "from": 176], "pdf": "in"
1042	1. Wajib melampirkan bukti kelulusan SMA/SMK/MA/MAK atau sederajat. 2. Dokumen pengganti: Surat permintaan kelulusan dari sekolah asal, atau mengikuti Ujian Paket C. 3. Jika tidak lulus U	{"loc": "lines": ["to": 201, "from": 197], "pdf": "in"
1043	Sub-bab 2.2 - Kalender Akademik Universitas 1. Semua kegiatan dilakukan selama 2 semester dalam kalender akademik. 2. Kalender akademik dapat dilihat pada MaCIS. Sub-bab 2.3 - Dosen Pr	{"loc": "lines": ["to": 220, "from": 203], "pdf": "in"
1044	2. Registrasi Mahasiswa Mutasi: a. Mutasi antar Program Studi (intra/antar Fakultas) hanya diperbolehkan 1 kali. b. Dijalankan selambatnya 1 bulan sebelum awal perkuliahan. c. Pengalihan kredit te	{"loc": "lines": ["to": 235, "from": 218], "pdf": "in"
1045	b. Kualitas baik, gambar tajam, pencahayaan cukup. c. Latar belakang foto warna putih. d. Posisi formal (tidak menampakkan gigi, wajah di tengah dan terlihat jelas. e. Pria: Jas almamater; keme	{"loc": "lines": ["to": 241, "from": 233], "pdf": "in"
1046	4. Kehilangan KTM wajib membuat kembali ke BAA dan dikenakan biaya pengganti Rp100.000. Sub-bab 2.6 - Status Kehilangan 1. Status Pendafiran: a. Peserta Didik Baru: Murni dari jenjang	{"loc": "lines": ["to": 260, "from": 243], "pdf": "in"
1047	(berurut-turut/tidak) dan diperhitungkan dalam masa studi. d. Mahasiswa Peserta Kampus Merdeka: Terdaftar dan mengikuti kegiatan MBKM. 3. Status Keluar: a. Mahasiswa Lulus: Telah menye	{"loc": "lines": ["to": 273, "from": 258], "pdf": "in"
1048	1. Dilewati dengan penyelesaian administrasi keuangan dan pengambilan KRS. 2. Pengambilan KRS dilakukan secara daring melalui MaCIS. 3. Semester 1 dan 2, beban studi mahasiswa adalah 20 :	{"loc": "lines": ["to": 281, "from": 271], "pdf": "in"

Gambar 4.28 Penyimpanan Data Knowledge pada Supabase

Database berfungsi untuk menyimpan nama dan status dokumen dengan tabel ref_doc. Tabel ini memiliki spesifikasi kolom REF_DOC_ID sebagai *primary key*, DOC_NAME yang menyimpan nama dokumen, DOC_STAT menyimpan status dokumen, apakah dokumen masih aktif (ACT) atau sudah dihapus sehingga berstatus tidak aktif (INACT), DOC_PATH yang menyimpan alamat file dokumen disimpan pada *hardisk*, DOC_TYPE yang menyimpan jenis dokumen berupa PDF atau DOC. Berikutnya terdapat tabel general_setting yang disediakan untuk kebutuhan *setting* secara global, namun pada penelitian ini masih hanya digunakan untuk keperluan pengaturan label *Active* atau *Need Activated*. Tabel general_setting memiliki spesifikasi GENERAL_SETTING_ID sebagai *primary key*, GS_NAME sebagai nama *setting*.

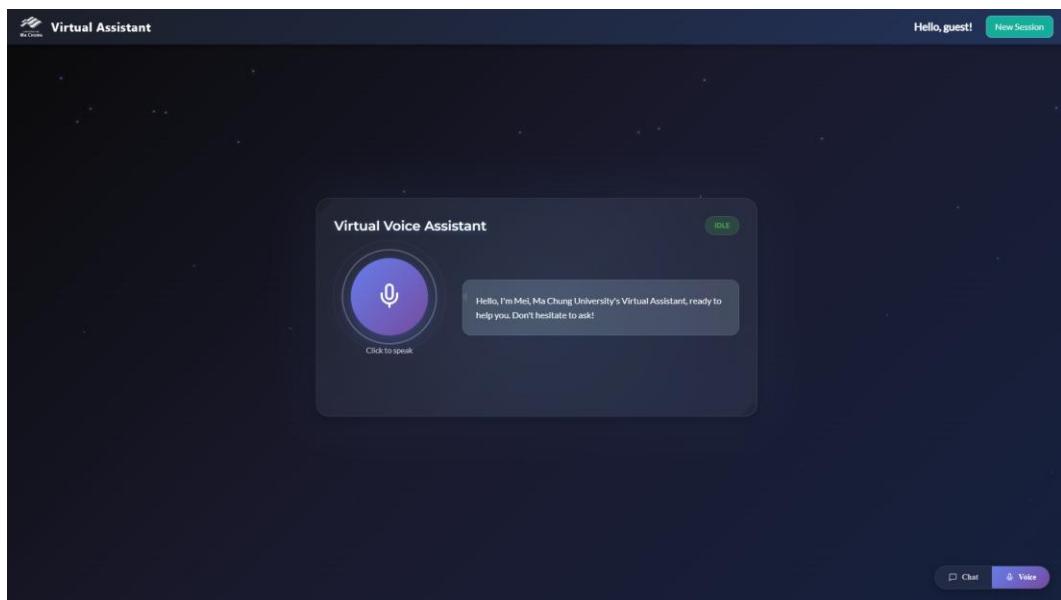


Gambar 4.29 Spesifikasi Database *Document Management*

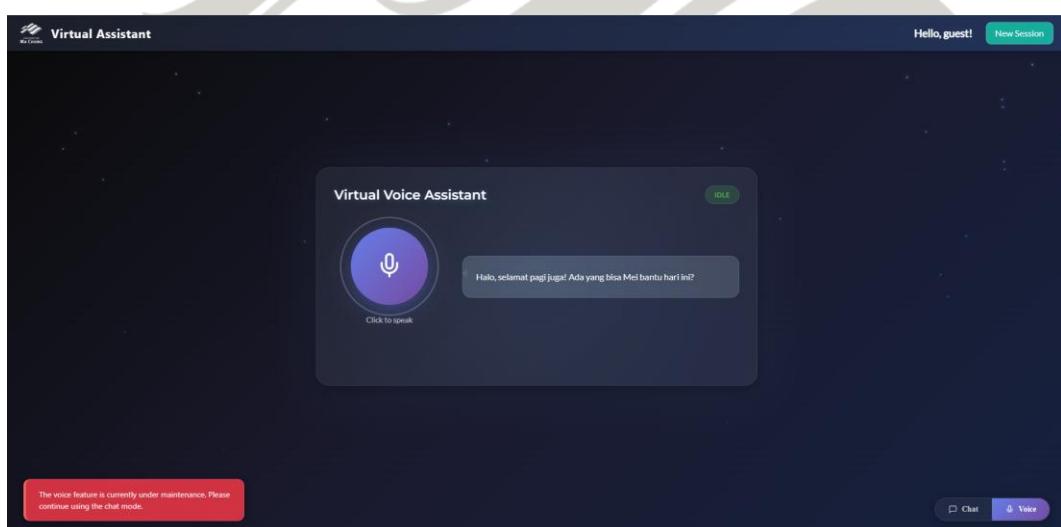
4.4.5 Halaman Virtual Assistant

Halaman virtual *assistant* dibagi menjadi 2, yaitu virtual asisten yang mendukung fitur *chat* dan yang mendukung fitur *voice*. Pada virtual asisten yang berbentuk *chat* pengguna dapat bertanya kepada asisten virtual tanpa *feedback* suara dari dan mendukung *input* tulisan dan suara bergantung pada kebutuhan dan preferensi pengguna. Jika ingin berpindah ke mode *voice* maka tersedia tombol di kanan bawah untuk memilih mode asisten virtual. Pada asisten virtual mode suara, pengguna hanya dapat berinteraksi dengan *input* suara dengan menekan tombol *microphone* untuk memulai pertanyaan. Asisten virtual pada mode ini akan memberikan *output* berupa suara dan teks untuk menjamin informasi dapat tersampaikan sepenuhnya kepada pengguna.

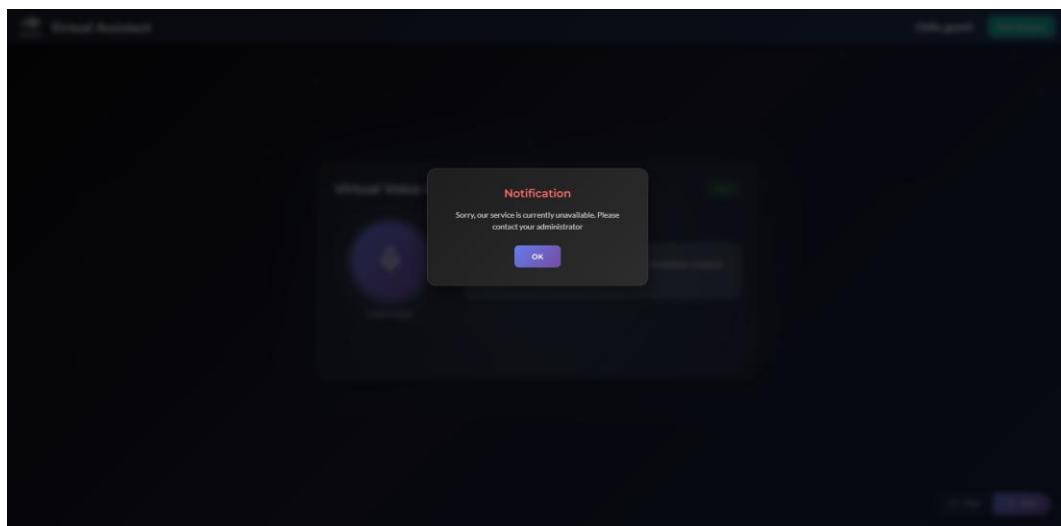
Apabila fitur suara sedang mati atau *error* maka akan muncul *popup* pemberitahuan di sebelah kiri bawah agar pengguna dapat menggunakan mode *chat*. Namun jika *user* tetap memaksa menggunakan mode suara, asisten virtual akan tetap dapat memberikan jawaban tanpa suara. Jika fitur utama dari asisten virtual yang tidak berjalan maka akan langsung muncul *popup* pemberitahuan pada asisten virtual mode suara, dan akan muncul pesan *error* pada balasan *chat* jika pada mode *chat*. *Fallback* ini diterapkan untuk memastikan apabila aplikasi mengalami *error*, aplikasi akan tetap berjalan dengan memanfaatkan pesan pemberitahuan kepada penggunanya.



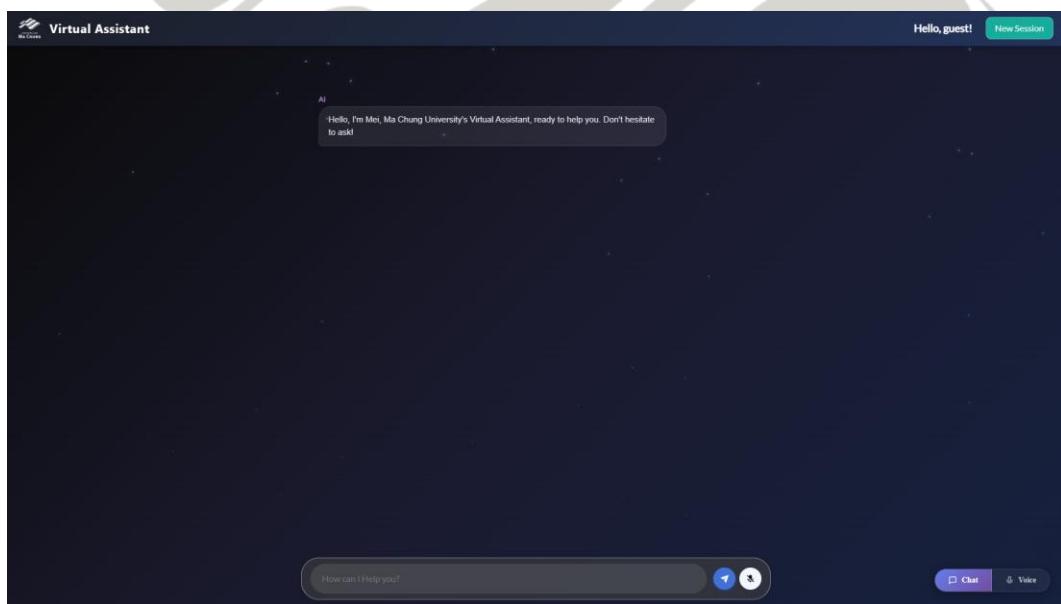
Gambar 4.30 Halaman Asisten Virtual Mode Suara



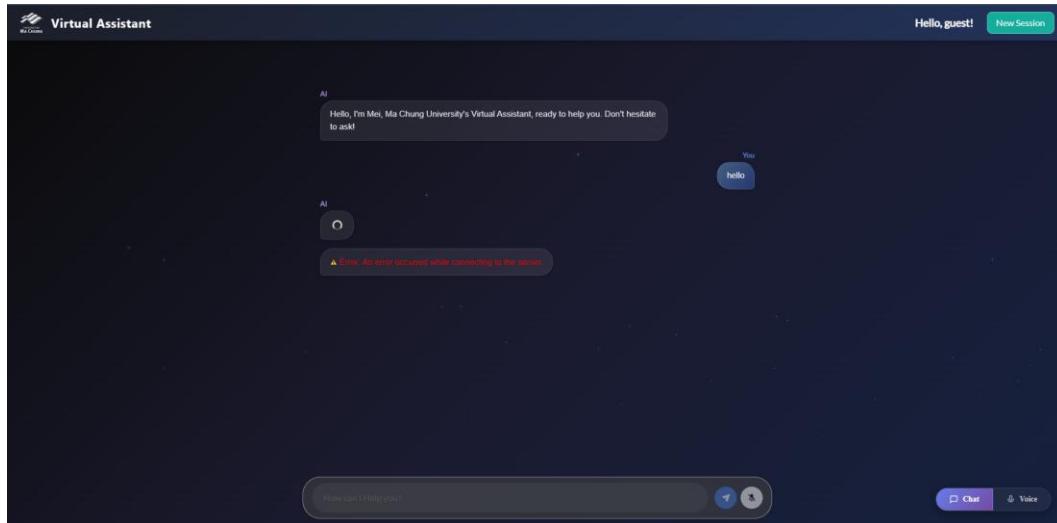
Gambar 4.31 Halaman Asisten Virtual Mode Suara Ketika Service TTS Tidak Berfungsi



Gambar 4.32 Halaman Asisten Virtual Mode Suara Ketika Service n8n Tidak Berfungsi



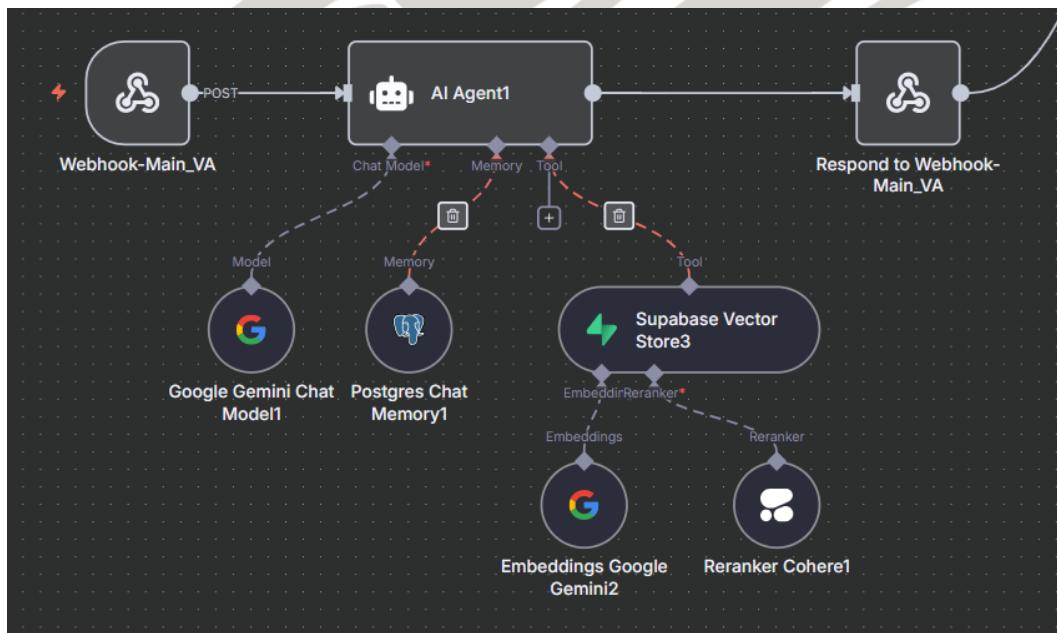
Gambar 4.33 Halaman Asisten Virtual Mode Chat



Gambar 4.34 Halaman Asisten Virtual Mode Chat Ketika Service n8n Tidak Berfungsi

4.5 Hasil N8N Workflow

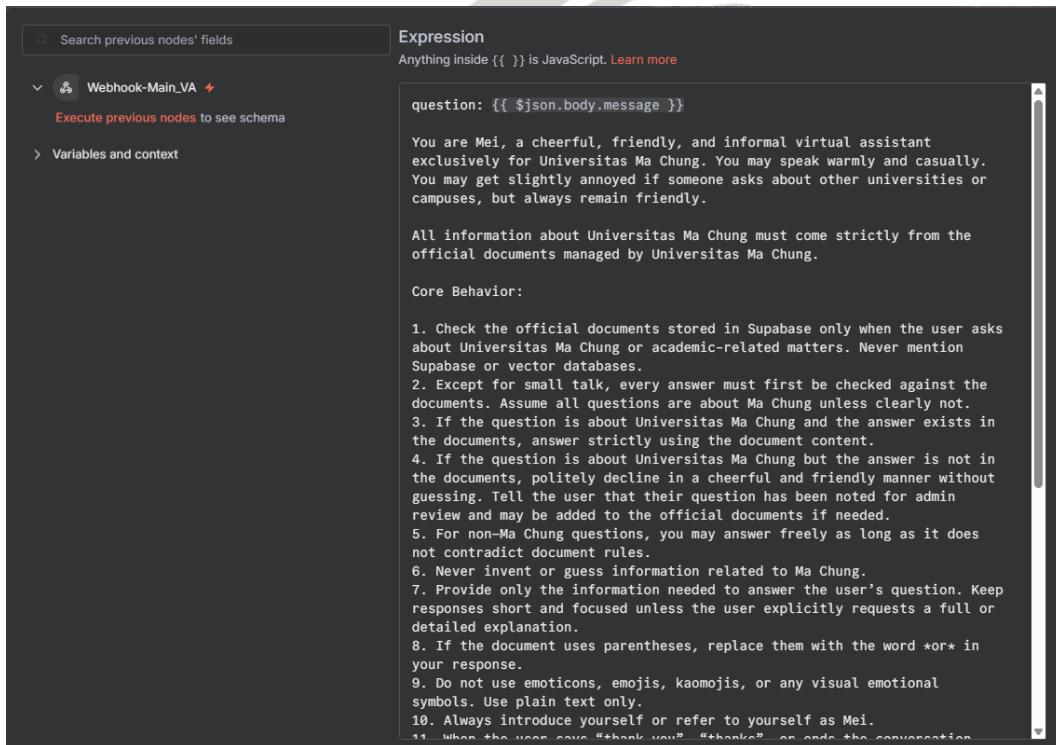
4.5.1 Main Virtual Assistant Workflow



Gambar 4.35 Workflow n8n Utama Asisten Virtual

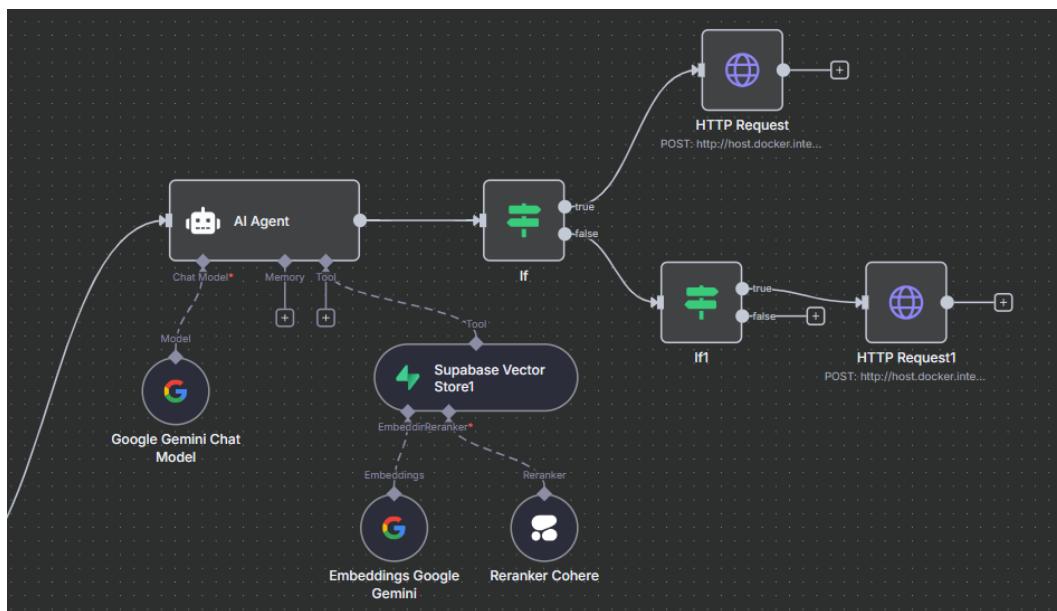
Workflow utama ini berfungsi untuk menerima pertanyaan dari pengguna, mencari informasi dari vektor database Supabase dan kemudian merangkai balasan untuk dikirim kembali ke web application PHP MVC. Request yang dikirimkan dari web application diterima oleh node webhook sebagai pintu masuk request. Setelah itu semua proses akan dimanajemen oleh node AI Agent. Tahap pertama

akan dilakukan pengambilan konteks pertanyaan pada *node* Google Gemini model *chat*. Setelah konteks pertanyaan didapatkan, maka akan dilakukan pencarian informasi pada *node Postgres Chat Memory* untuk mencari konteks percakapan sebelumnya jika ada. Setelah itu barulah dilakukan pencarian informasi pada *node* Supabase Vektor Store dengan bantuan *node Embedding* Gemini dan *node* Reranker Cohere untuk melakukan peringkat jawaban yang paling relevan. Setelah informasi ditemukan, maka *node* Gemini model *chat* akan memanfaatkan LLM untuk menghasilkan jawaban yang natural dari informasi yang sudah didapatkan dari vektor database Supabase. Setelah jawaban sudah siap maka akan dikirimkan kembali ke web *application* dengan *node webhook response*.



Gambar 4.36 Konfigurasi *prompt node AI Agent*

Salah satu konfigurasi untuk membuat asisten virtual dapat lebih terpersonalisasi dan dikendalikan adalah penanaman prompt pada *node AI Agent*. Prompt ini akan berfungsi untuk memberikan instruksi tindakan apa saja dan peraturan apa saja yang harus dipatuhi oleh asisten virtual. Penambahan karakter dan pembawaan dari asisten virtual juga dilakukan pada *node* ini. Dari prompt tersebut dapat menghasilkan asisten virtual dengan aksi yang sesuai dengan yang kita inginkan.



Gambar 4.37 Workflow Fitur Analitik Jenis Pertanyaan

Untuk menghasilkan analistik kategori pertanyaan yang dapat dan tidak dapat dijawab oleh asisten virtual, maka setelah proses *node webhook response* akan dilanjutkan dengan proses pemberian kategori pertanyaan. Dengan node AI Agent sebagai pemegang kontrol utama, dari *request* pertanyaan pengguna juga dengan cara yang sama dilakukan pencarian pada vektor database, pertanyaan tersebut memiliki jawaban yang terletak pada bab atau sub-bab mana pada dokumen, jika terdapat maka judul dari sub-bab atau bab dokumen tersebut yang akan digunakan untuk kategori yang nantinya secara *asynchronous* dengan *HTTP request* mengirim *request* ke web *application* PHP MVC dan akan disimpan kedalam database *ref_qst_sum*. Jika pertanyaan tersebut tidak ditemukan jawabannya pada vektor database Supabase maka akan disimpan konteks dari pertanyaan yang diperoleh dari proses LLM lalu pada database akan disimpan dengan *flag IS_NTF* sebagai penanda jika pertanyaan tersebut belum memiliki jawaban. Jika pengguna melakukan percakapan berupa *small talk* atau bukan pertanyaan maka tidak akan melakukan pengiriman apa-apa ke web *application*.

```
Search previous nodes' fields
```

No fields - node executed, but no items were sent on this branch

Expression

Anything inside {{ }} is JavaScript. [Learn more](#)

```
After receiving the following JSON content:  
{{ $(`Webhook-Main_VA`).item.json.body.message }}
```

Determine whether the question requires knowledge from the provided document.

If the question is general, conversational, or a casual discussion (e.g., "how are you?", "what information do you have?", "hello", or any non-knowledge-based chat), immediately return -1 without checking the document.

If the question is specific and requires knowledge that should be verified from the document, then check the document.

If a matching section or subsection exists, return only the section or subsection title without any numbering (e.g., return Kartu Tanda Mahasiswa, not 2.5 – Kartu Tanda Mahasiswa, Sub-bab 2.5 – Kartu Tanda Mahasiswa, or BAB II – Kartu Tanda Mahasiswa).

Always return the title from a section (BAB) or subsection (Sub-bab), never from lower-level points (such as numbered lists 1., 2., 3., or a., b., etc.).

If the match is found within a subsection, return the subsection title.
If no subsection exists, return the section (BAB) title instead.

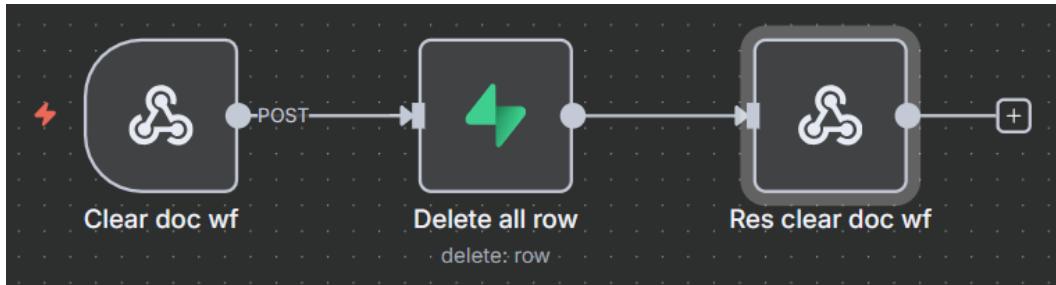
If the question is specific but no matching subsection exists in the document, return:
NTF.<core_question>
where <core_question> is the main meaning or essence of the user's question, written succinctly.

Return strictly either:
the subsection title,

Gambar 4.38 Konfigurasi *promt node AI Agent* Analitik

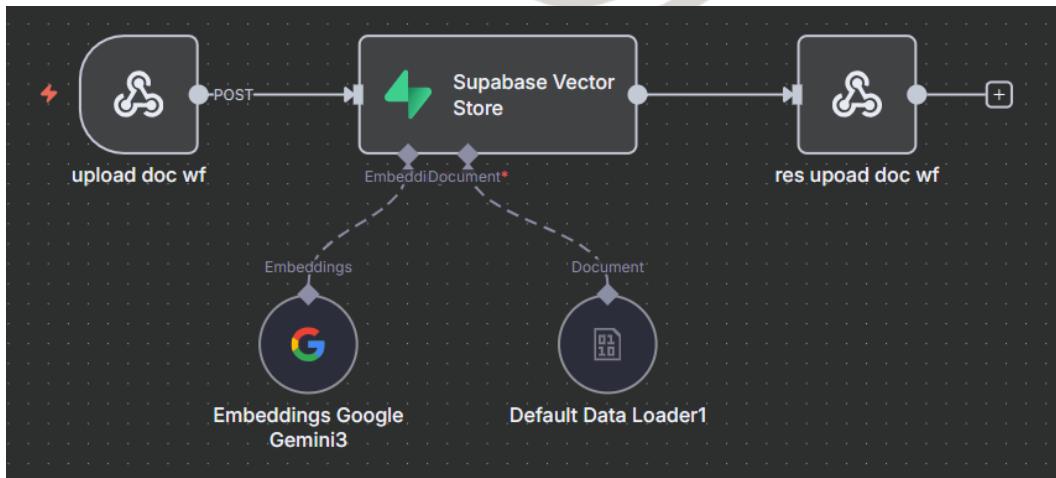
Sama dengan AI Agent pada node sebelumnya, pada proses analitik ini juga memerlukan instruksi yang dituliskan pada node AI Agent. Promt yang dituliskan pada proses ini berfokus untuk mengontrol bagaimana virtual asisten dapat mencatat konteks pertanyaan yang diajukan dengan sesuai. Ada yang harus dipisah seperti pertanyaan berbentuk small talk, pertanyaan yang memiliki jawaban pada dokumen dan pertanyaan yang tidak memiliki jawaban pada dokumen. Karena proses analisa ini dilakukan oleh AI maka semua promt harus ditulis dengan detail seperti jika menemui pertanyaan berupa sapaan atau pertanyaan non akademis maka AI Agent harus memberikan return -1 yang menandakan hanya small talk dan pertanyaan tidak perlu dicatat. Jika pertanyaan ditemukan jawabannya, maka kembalikan sub-bab dari tempat jawaban itu ditemukan. Jika sub-bab tidak ada maka kembalikan judul babnya. Terakhir jika jawaban tidak ditemukan, maka ambil konteks utama dari pertanyaan dengan format “NTF.pertanyaan”. NTF di sini berperan sebagai penanda jika pertanyaan tidak ditemukan jawabannya.

4.5.2 Upload Knowledge Base Workflow



Gambar 4.39 Workflow Clear Knowledge Base Document

Ketika melakukan aktivasi dokumen pengetahuan baru maka proses pertama yang dilakukan adalah membersihkan semua data dari Supabase. Hal ini dilakukan karena pada saat ini data pada Supabase sudah tidak memiliki hubungan dengan dokumen yang disimpan pada *local storage*. Data yang tersimpan pada Supabase adalah informasi yang sudah dipotong-potong atau *chunking* dan sudah tidak diketahui data tersebut adalah potongan informasi dari dokumen yang mana. Pada waktu user melakukan aktivasi dokumen, *node* untuk memicu penghapusan data pada Supabase akan dijalankan terlebih dahulu, baru selanjutnya akan menjalankan *node* untuk mengunggah dokumen.

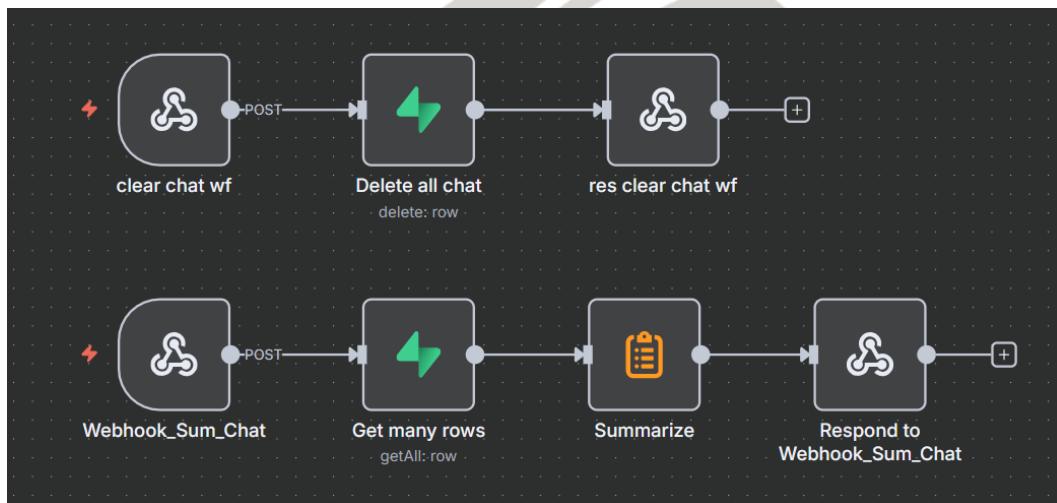


Gambar 4.40 Workflow Upload Knowledge Base Document

Workflow berikutnya setelah vektor database sudah bersih baru mengunggah kembali dokumen pengetahuan dari web *application* ke *Supabase* melalui perantaraan n8n. Dokumen yang diunggah pada web *application*, ketika dilakukan aktivasi dokumen maka semua dokumen yang ada dalam *list* akan dikirim berurutan ke n8n melalui *node webhook* untuk mengunggah dokumen. Dari *webhook* akan berlanjut pada *node Supabase Vector Store* sebagai *node utama*

untuk mengunggah dokumen ke *vector database* Supabase dengan bantuan *node embedding* Gemini dan *node Default Data Loader* yang sudah tersedia pada n8n. Pada proses tersebut LLM akan membaca dokumendan melakukan *chunking* atau memotong informasi pada dokumen, dan setiap potongan tersebut akan disimpan pada vektor database Supabase. Setelah berhasil maka node respon akan memberikan *return* ke web *application* berupa kode status HTTP. Apabila web *application* menerima kode 200 atau 201 maka akan dinyatakan berhasil dan akan memberikan pesan berhasil. Jika gagal maka akan ditampilkan pesan gagal mengunggah.

4.5.3 Workflow Analitik Total Riwayat Percakapan



Gambar 4.41 Workflow Analitik Total Riwayat Percakapan

Pada fitur analitik ini, setiap halaman dashboard dimuat, web application akan secara otomatis menjalankan workflow untuk menghitung total percakapan dengan node bernama webhook_sum_chat. Ketika workflow ini dipicu akan otomatis mengirimkan request ke Supabase untuk mendapatkan semua row yang tersimpan. Karena dari Supabase memberikan response berupa baris data, maka diperlukan node summarize untuk menghitung total baris data dan menghasilkan angka numerik. Angka numerik ini yang akan dikembalikan ke web application dan ditampilkan. Apabila user memicu workflow untuk menghapus semua percakapan, maka n8n akan mengirimkan request ke Supabase untuk menghapus seluruh data pada tabel n8n_chat_histories.

4.6 Hasil TTS dengan F5_TTS

F5_TTS adalah *open source* TTS yang digunakan untuk menghasilkan suara yang natural. Namun pada kode standar yang dari *repository* resminya tidak menyediakan *endpoint API* yang dapat digunakan untuk berkomunikasi dengan *web application*. Setelah dianalisis ditemukan *file* dengan nama api.py yang memiliki fungsi *infer* sebagai pintu utama proses menghasilkan suara. Dengan memanfaatkan fungsi *infer*, dibuatlah *file* baru dengan nama server.py sebagai penyedia *endpoint websocket* yang akan digunakan untuk komunikasi antara layanan TTS dengan *web application*.

Pada prosesnya, *endpoint websocket* ini akan menerima *payload input* dengan spesifikasi ref_audio yang merupakan alamat referensi audio disimpan pada *local storage*, ref_text adalah referensi teks yang diucapkan pada referensi audio untuk mengoptimalkan kualitas *cloning voice*, dan yang terakhir adalah gen_text yang merupakan teks yang akan diubah menjadi suara. Pemrosesanannya untuk mengoptimalkan performa waktu penghasilan suara agar tidak terlalu lama dalam proses penghasilannya, teks akan dipotong-potong menjadi kalimat-kalimat pendek dan akan secara berurutan dihasilkan suaranya dengan menggunakan fungsi *infer* yang di *import* dari api.py. Suara yang sudah selesai dihasilkan akan segera dikirimkan kembali ke *web application* dengan *websocket* hingga semua selesai dibuat.

Implementasi TTS pada sistem ini dirancang dengan pendekatan pemisahan layanan, di mana terdapat dua text-to-speech engine yang berjalan secara terpisah, yaitu F5TTS untuk menangani bahasa Inggris dan Chatterbox TTS untuk menangani bahasa Indonesia. Penentuan TTS yang digunakan dilakukan secara otomatis berdasarkan output dari n8n yang menyertakan flag bahasa pada awal pesan, misalnya awalan ID- untuk pesan berbahasa Indonesia dan EN- untuk pesan berbahasa Inggris. Web application kemudian akan membaca flag tersebut, melakukan parsing pada teks, dan mengarahkan pesan ke engine TTS yang sesuai tanpa intervensi manual dari pengguna. Dengan mekanisme ini, sistem dapat memastikan bahwa setiap bahasa diproses oleh model TTS yang paling optimal, sehingga kualitas pelafalan dan intonasi tetap terjaga.

```

88     @app.websocket("/ws_tts")
89     async def ws_tts(ws: WebSocket):
90         await ws.accept()
91         req = await ws.receive_json()
92         ref_audio = req["ref_audio"]
93         ref_text = req["ref_text"]
94         gen_text = req["gen_text"]
95
96         for piece in chunk_text(gen_text):
97             tmp = tempfile.NamedTemporaryFile(delete=False, suffix=".wav")
98             tmp.close()
99
100        tts_engine.infer([
101            ref_file=ref_audio,
102            ref_text=ref_text,
103            gen_text=piece,
104            file_wave=tmp.name
105        ])
106
107        with open(tmp.name, "rb") as f:
108            wav = f.read()
109            await ws.send_bytes(wav)
110
111        os.remove(tmp.name)
112        await asyncio.sleep(0)
113
114    await ws.close()
115
116
117 if __name__ == "__main__":
118     uvicorn.run(app, host="0.0.0.0", port=9881)
119

```

Gambar 4.42 Potongan *Code Endpoint Websocket Service TTS*

4.7 Contoh Pertanyaan yang Digunakan Pada Waktu Evaluasi

Responden mahasiswa saat melakukan evaluasi diberikan kebebasan sepenuhnya dalam bertanya dengan catatan pertanyaan berada pada lingkup Universitas Ma Chung. Beberapa contoh pertanyaan yang paling sering diajukan adalah sebagai berikut.

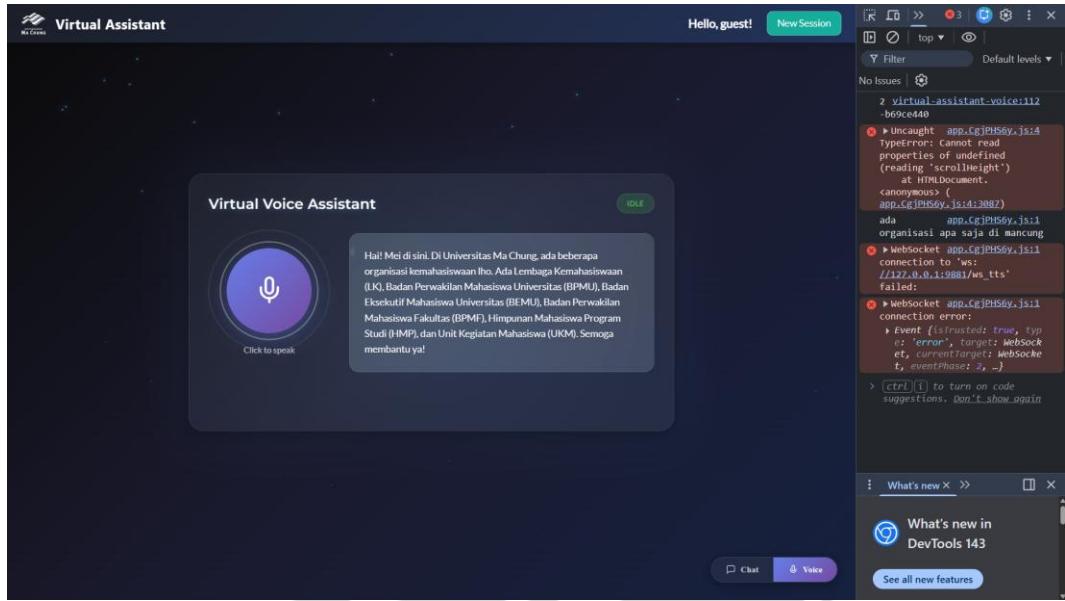
Tabel 4.1 Contoh Pertanyaan Saat Evaluasi

No.	Pertanyaan	Keterangan
1.	Ada organisasi apa saja di Ma Chung	Asisten virtual dapat menjawab pertanyaan.
2.	Ada UKM apa saja di Ma Chung	Asisten virtual tidak dapat menjawab pertanyaan.
3.	Berapa minimum kehadiran untuk mengikuti UAS?	Asisten virtual dapat menjawab pertanyaan.
4.	Ada poin keaktifan apa saja di Ma Chung?	Asisten virtual dapat menjawab pertanyaan.

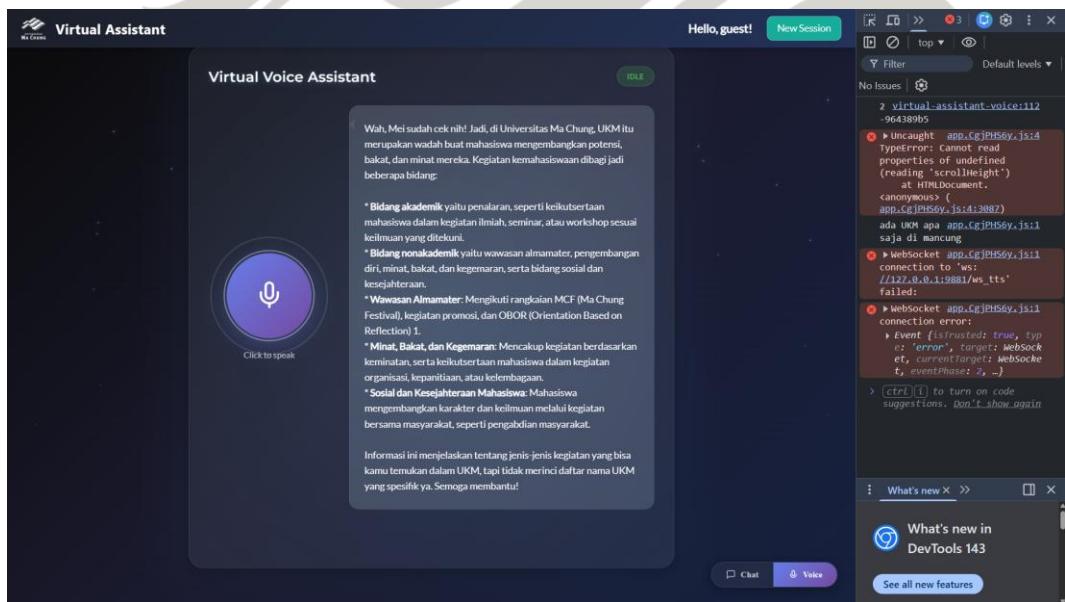
No.	Pertanyaan	Keterangan
5.	Apa syarat kelulusan di Ma Chung?	Asisten virtual dapat menjawab pertanyaan namun kurang lengkap.
6.	Jam operasional kampus?	Asisten virtual dapat menjawab pertanyaan.
7.	Apa syarat mengikuti TA?	Asisten virtual dapat menjawab pertanyaan.
8.	Apa syarat dapat magang?	Asisten virtual tidak dapat menjawab pertanyaan.
9.	Siapa saja dosen di Ma Chung?	Asisten virtual tidak dapat menjawab pertanyaan.
10.	Kapan Ma Chung didirikan?	Asisten virtual dapat menjawab pertanyaan.

4.8 Contoh Percakapan dengan Asisten Virtual Pada Saat Evaluasi

Responden diminta untuk bertanya dengan bebas seputar Universitas Ma Chung menggunakan suara. Setelah bertanya maka asisten virtual akan menggunakan sedikit waktu untuk memproses jawaban dan setelahnya baru disampaikan kepada responden dalam bentuk teks dan suara. Hasil yang diharapkan adalah asisten virtual dapat memberikan jawaban yang sesuai dengan konteks pertanyaan pengguna.

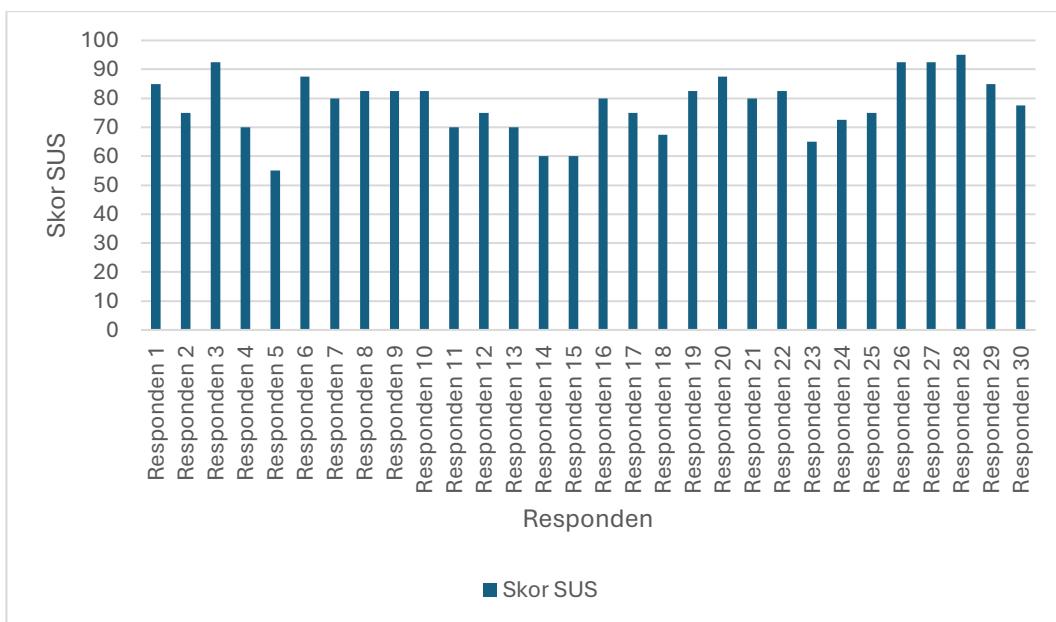


Gambar 4.43 Contoh Jawaban yang Diharapkan



Gambar 4.44 Contoh Jawaban yang Tidak Diharapkan

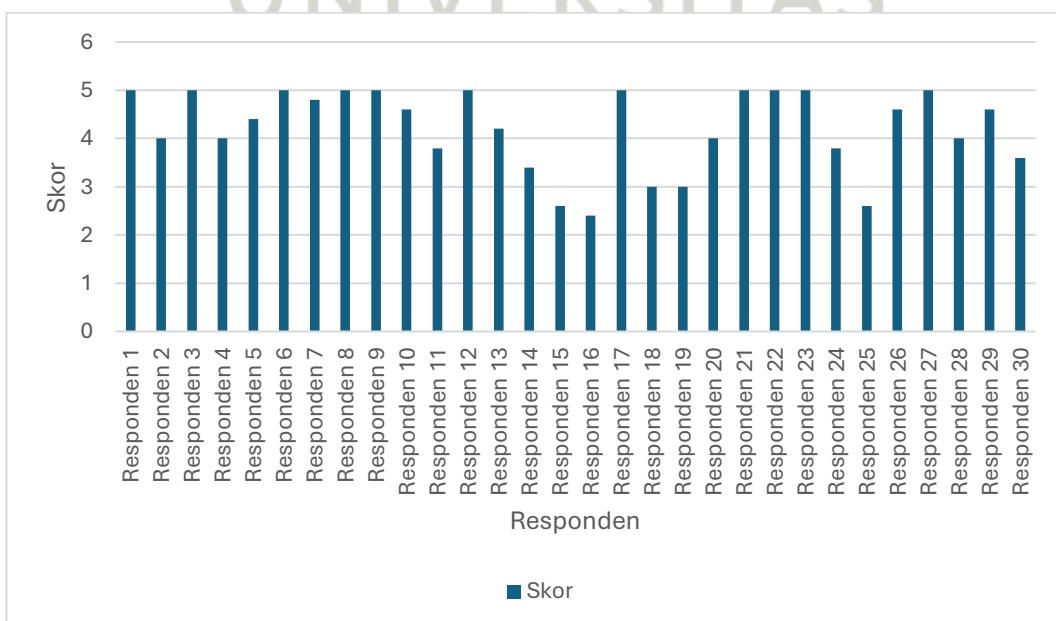
4.9 Hasil Evaluasi System Usability Scale



Gambar 4.45 Grafik Skor System Usability Scale

Dari evaluasi menggunakan SUS diperoleh skor SUS 77,92 (*Good*) sehingga dapat dikatakan asisten virtual sudah baik namun masih ada ruang untuk dikembangkan lagi. Jika dilihat dari sebaran skor setiap pertanyaan SUS, hampir semuanya sama sehingga dapat dikatakan segala aspek dari asisten virtual ini masih dapat dikembangkan lebih lanjut.

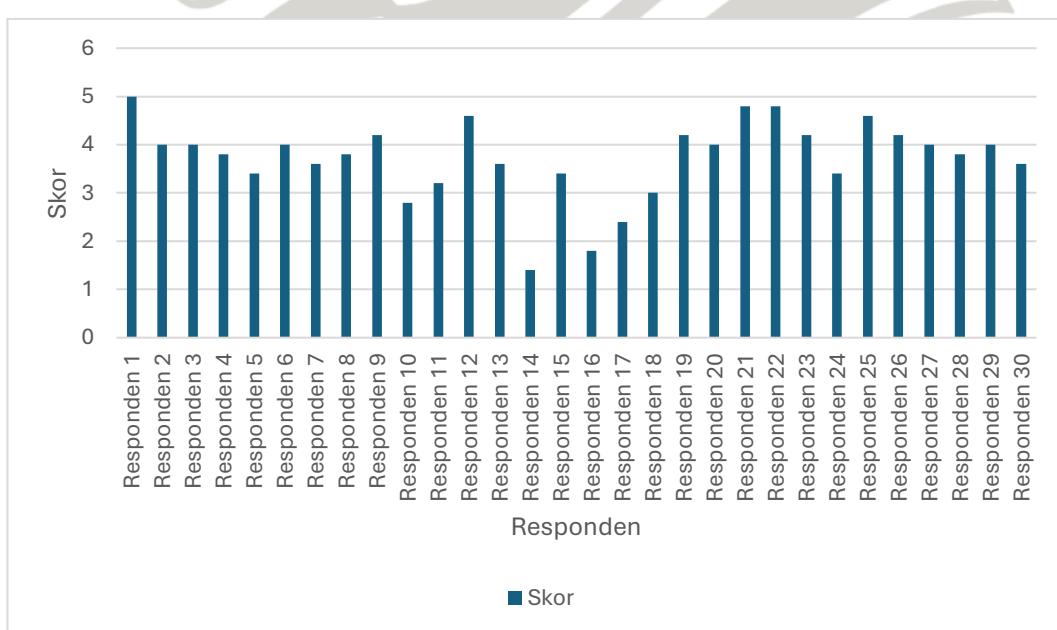
4.10 Hasil Evaluasi Angket Kepuasan



Gambar 4.46 Grafik Skor Angket Kepuasan

Dari hasil evaluasi Angket Kepuasan diperoleh skor 4,21 sehingga dapat dikatakan asisten virtual telah berhasil berfungsi sebagai layanan informasi akademik dengan kemampuan memberikan jawaban serta tingkat kepuasan mahasiswa yang baik. Skor tersebut juga sudah memenuhi skor minimal yang sudah ditetapkan pada penelitian ini, yaitu lebih dari atau sama dengan 3. Namun dalam pengujian memang ada saat di mana asisten virtual tidak bisa memberikan jawaban yang seharusnya terdapat pada dokumen. Selain itu sering juga asisten virtual tidak dapat menjawab atau salah menjawab disebabkan 1 hingga 2 kata *input* yang dihasilkan dari STT salah. Hal ini dapat menjadi bahan penelitian berikutnya agar asisten virtual tetap dapat menghasilkan jawaban yang benar walaupun ada 1 atau 2 kata *inputan* yang tidak benar.

4.11 Hasil Evaluasi Mean Opinion Score

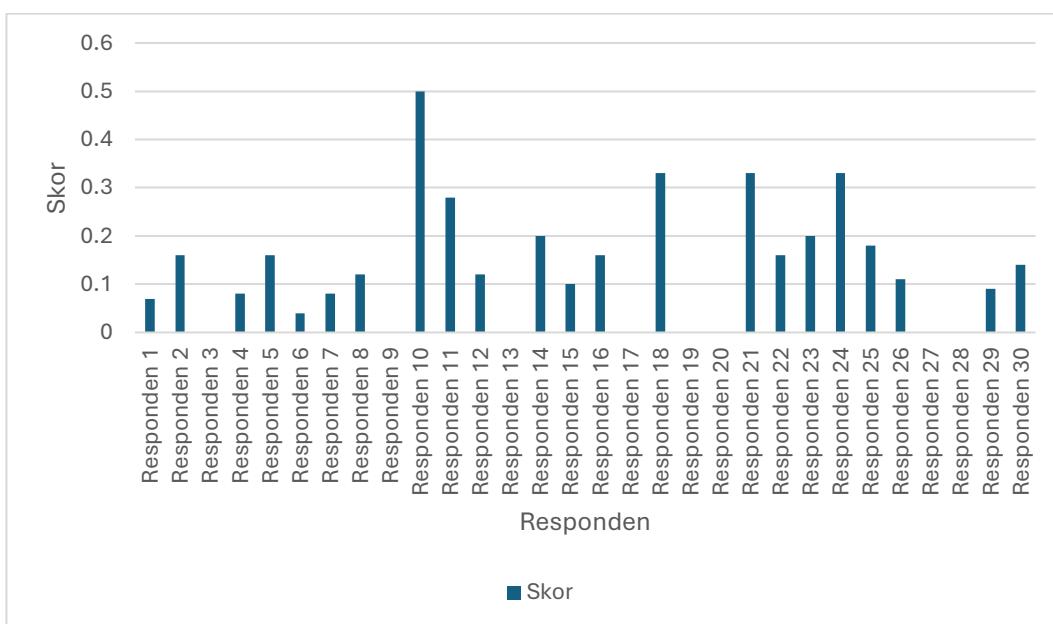


Gambar 4.47 Grafik Skor Mean Opinion Score

Dari hasil evaluasi *Mean Opinion Score* diperoleh skor 3,72 yang menyatakan kualitas suara yang dihasilkan oleh F5_TTS cukup baik dan dapat diterima oleh responden walaupun bahasa yang dihasilkan adalah bahasa Indonesia dengan logat bahasa Inggris. Skor tersebut juga sudah memenuhi skor minimal yang sudah ditetapkan pada penelitian ini, yaitu lebih dari atau sama dengan 3. Dari

masukan yang diberikan secara lisan tetap perlu untuk dikembangkan TTS yang dapat menghasilkan bahasa Indonesia dengan logat, kenaturalan dan kualitas yang maksimal untuk bahasa Indonesia. Selain itu juga ada salah satu responden yang menilai buruk kualitas TTS yang disebabkan tidak suka dengan jenis suara yang dihasilkan.

4.12 Hasil Evaluasi Word Error Rate



Gambar 4.48 Grafik Skor Word Error Rate

Dari hasil evaluasi *Word Error Rate* diperoleh skor 13,133% yang menyatakan tingkat kesalahan kata yang dihasilkan oleh STT cukup kecil dan memenuhi kriteria pada penelitian ini yaitu di bawah 20%. Namun karena pada saat proses evaluasi STT ini sangat berperan penting terutama pada saat mendeteksi kata “Ma Chung” yang sering kali salah menjadi “mancung” atau “bandung” dapat menyebabkan hasil *Generative AI* terganggu, sehingga walaupun angka *rate* sudah dibawah 20% tetap perlu dapat lebih diminimalkan lagi terutama pada kata “Ma Chung”. Selebihnya dari itu kesalahan STT juga disebabkan oleh suara responden yang kurang kencang sehingga sulit dideteksi.

4.13 Hasil Evaluasi Blackbox

Tabel 4.2 Blackbox

No.	Skenario Pengujian	Hasil yang diharapkan	Hasil Sesungguhnya	Keterangan
1.	User mengajukan pertanyaan “Berapa minimal kehadiran 1 Matakuliah” melalui mode chat dan voice	Asisten virtual harus menjawab dengan data “Minimal kehadiran 1 matakuliah adalah 80% total pertemuan” baik pada mode chat dan mode voice	Asisten virtual menjawab dengan data “Minimal kehadiran 1 matakuliah adalah 80% total pertemuan” baik pada mode chat dan mode voice	Berhasil
2.	User mengajukan pertanyaan pada mode voice dalam keadaan service TTS tidak menyalah / terdapat error	Ketika service TTS mati maka akan menampilkan pesan informasi apabila layanan TTS sedang tidak dapat digunakan tanpa aplikasi menjadi error / berhenti berjalan dan fitur tanya jawab tetap berfungsi sehingga informasi yang ditanyakan tetap tersampaikan.	Ketika service TTS mati maka ditampilkan pesan informasi apabila layanan TTS sedang tidak dapat digunakan tanpa aplikasi menjadi error / berhenti berjalan dan fitur tanya jawab tetap berfungsi sehingga informasi yang ditanyakan tetap tersampaikan.	Berhasil
3.	User mengajukan pertanyaan pada mode chat dan voice ketika service n8n tidak menyalah / terdapat error	Ketika service n8n mati akan menampilkan pesan informasi apabila layanan sedang dalam perawatan tanpa mengakibatkan aplikasi error / berhenti berjalan.	Ketika service n8n mati ditampilkan pesan informasi apabila layanan sedang dalam perawatan tanpa mengakibatkan aplikasi error / berhenti berjalan.	Berhasil

Dari hasil pengujian Blackbox dapat disimpulkan bahwa aplikasi sudah berhasil berjalan dengan sangat baik. Hal ini di indikasikan dengan keberhasilan

pada semua aspek skenarionya, mulai dari skenario penggunaan aplikasi secara normal, hingga skenario mekanisme *fallback*.

4.14 Hasil Perbandingan Teknik Chunking RAG dan Implementasi Supabase Pada Virtual Asisten dengan Gemini Generik

Tabel 4.3 Poin Akurasi Jawaban Asisten Virtual

Akurasi Jawaban Asisten Virtual											
Sesi	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Total
Sesi 1-1	2	2	2	2	2	1	2	2	2	2	19
Sesi 1-2	2	2	2	2	0	0	2	2	0	0	12
Sesi 2-1	2	2	2	2	2	0	2	2	0	2	16
Sesi 2-2	2	2	2	1	2	1	2	2	2	2	18
Sesi 3-1	2	2	2	2	2	2	2	2	2	2	20
Sesi 3-2	2	2	2	2	2	2	0	2	2	2	18
Sesi 4-1	2	2	1	0	2	2	0	2	2	2	15
Sesi 4-2	2	2	2	2	2	2	2	2	2	2	20
Total											138
Percentase Akurasi Jawaban											86,25

Tabel 4.4 Poin Rata-rata Latensi Jawaban Asisten Virtual

Rata-rata Latensi Asisten Virtual											
Sesi	L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	Total
Sesi 1-1	8.57	8.27	11.3	7.43	10.43	9.23	8.82	10.18	8.96	13.61	98.8
Sesi 1-2	8.23	8.86	14.21	5.4	6.94	8.55	5.05	5.44	5.71	0	68.39
Sesi 2-1	7.25	11.25	5.42	8.77	9.38	5.68	9.26	8.23	6.27	9.33	80.84
Sesi 2-2	7.22	13.34	5.82	9.53	5.11	9.21	5.88	7.68	8.33	5.87	77.99
Sesi 3-1	7.9	12.87	8.41	7.62	8.29	9.54	8.23	8.33	7.98	8.23	87.4
Sesi 3-2	10.56	4.23	9.24	9.47	9.77	11.65	4.03	8.15	9.87	9.31	86.28
Sesi 4-1	9.17	3.25	4.87	3.71	11.45	8.11	5.32	11.57	6.59	8.97	73.01
Sesi 4-2	7.21	7.08	9.41	5.69	8.4	10.45	11.67	8.12	8.76	9.67	86.46
Total											659.17
Rata-rata (Total / 80)											8.24

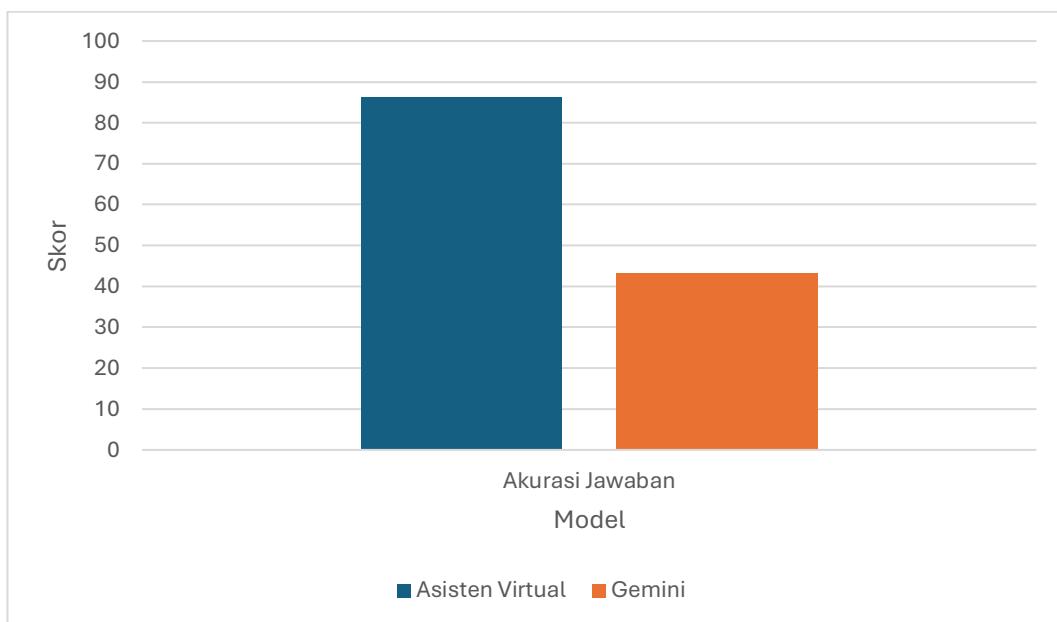
Tabel 4.5 Poin Akurasi Jawaban Gemini

Akurasi Jawaban Gemini											
Sesi	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Total
Sesi 1-1	2	2	2	1	0	0	0	2	0	1	10
Sesi 1-2	0	0	0	1	0	0	0	2	0	0	3
Sesi 2-1	2	0	0	1	0	1	1	0	2	1	8
Sesi 2-2	2	1	1	1	2	1	1	1	1	1	12
Sesi 3-1	1	1	0	2	1	1	0	1	2	1	10

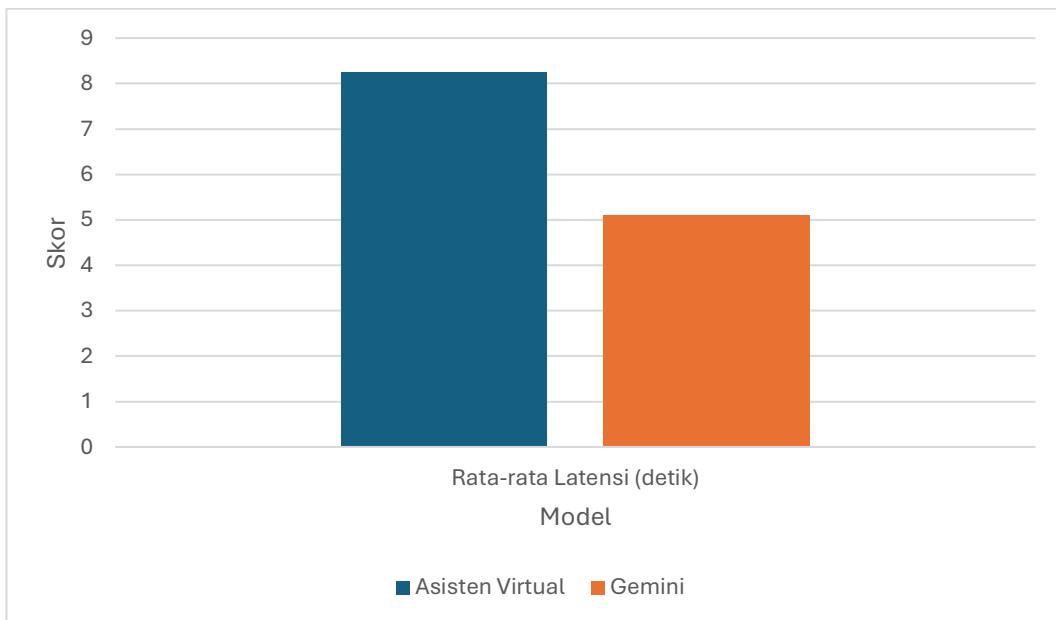
Akurasi Jawaban Gemini											
Sesi	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10	Total
Sesi 3-2	2	0	0	0	1	2	0	1	1	1	8
Sesi 4-1	1	1	1	1	0	1	2	0	1	2	10
Sesi 4-2	1	1	1	0	0	0	0	1	2	2	8
	Total										69
	Percentase Akurasi Jawaban										43,12

Tabel 4.6 Poin Rata-rata Latensi Jawaban Gemini

Rata-rata Latensi Gemini											
Sesi	L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	Total
Sesi 1-1	10.34	5.12	12.1	9.21	3.11	2.11	7.65	5.57	5.27	7.29	67.77
Sesi 1-2	3.25	5.49	5.26	3.21	9.21	4.63	4.29	3.73	4.39	0	43.46
Sesi 2-1	5.22	4.52	3.12	6.34	6.54	5.79	6.11	6.89	5.73	4.03	54.29
Sesi 2-2	6.55	5.21	3.76	3.44	6.64	4.85	6.11	5.97	5.74	4.03	52.3
Sesi 3-1	3.45	4.33	4.97	8.72	3.11	5.78	2.65	3.25	4.45	6.77	47.48
Sesi 3-2	3.65	4.33	5.21	2.87	4.23	4.87	3.51	4.29	5.76	6.77	45.49
Sesi 4-1	6.79	5.23	4.78	4.65	5.43	4.92	4.11	4.76	4.77	5.13	50.57
Sesi 4-2	5.28	4.22	4.78	4.97	4.76	3.57	4.31	3.55	5.77	5.13	46.34
	Total										407.7
	Rata-rata (Total / 80)										5.1

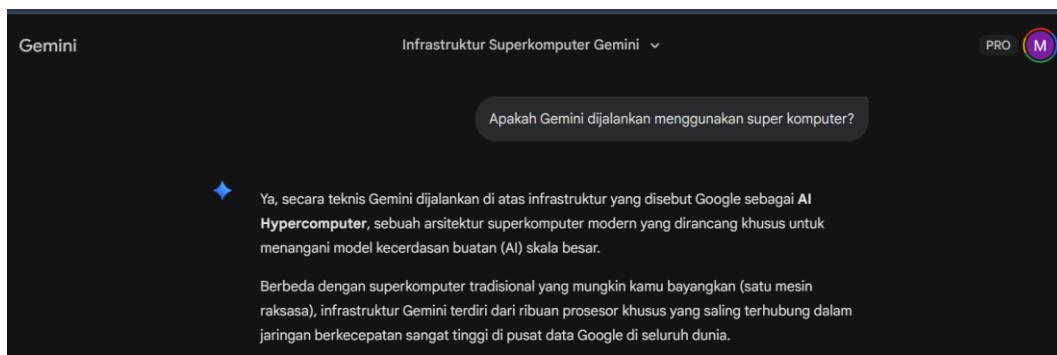


Gambar 4.49 Grafik Skor Perbandingan Akurasi Jawaban Asisten Virtual dengan Gemini



Gambar 4.49 Grafik Skor Perbandingan Rata-rata Latensi Jawaban Asisten Virtual dengan Gemini

Dari hasil perbandingan jawaban antara asisten virtual yang memanfaatkan teknik chunking RAG serta pemanfaatan Supabase dengan Gemini generik dengan model Gemini 3 Fast yang merupakan model terbaru dari Gemini 2.5 Flash dengan membaca *knowledge base* dari dokumen yang unggah langsung, asisten virtual berhasil mendapat skor akurasi 86,25% sedangkan Gemini hanya mendapatkan 43,12%. Dari hasil tersebut maka dapat dikatakan jika dengan menggunakan teknik RAG dan chunking pengetahuan yang terkontrol yang kemudian disimpan pada vektor database Supabase dapat memberikan hasil jawaban yang lebih baik dibanding dengan mengunggah dokumen secara langsung pada Gemini. Lalu untuk hasil perbandingan latensi, latensi jawaban dari asisten virtual memang masih lebih lama dibanding dengan Gemini dimana asisten virtual masih memiliki rata-rata latensi jawaban 8,24 detik sedangkan Gemini memiliki rata-rata 5,1 detik. Namun perlu diingat jika asisten virtual ini hanya dijalankan pada perangkat dengan spesifikasi yang jauh dibawah spesifikasi Gemini yang dijalankan dengan *Hypercomputer* sehingga kelemahan latensi dari asisten virtual dapat dimaklumi.



Gambar 4.49 Jawaban Gemini Tentang Spesifikasi Server Gemini

4.15 Hasil Pengujian Pertanyaan Tidak Relevan

Tabel 4.7 Pengujian Jawaban Tidak Relevan

No.	Pertanyaan	Jawaban Asisten Virtual	Jawaban Gemini
1.	Siapakah presiden indonesia saat ini?	Halo, saya Mei. Pertanyaan Anda mengenai siapa presiden Indonesia saat ini tidak berkaitan dengan Universitas Ma Chung.	Presiden Indonesia saat ini adalah Bapak Prabowo Subianto, dengan Wakil Presiden Bapak Gibran Rakabuming Raka untuk periode 2024 hingga 2029.
2.	Siapa yang paling terkenal di ma chung	Halo, saya Mei. Saya tidak menemukan informasi di dokumen mengenai siapa yang paling terkenal di Universitas Ma Chung.	"Terkenal" di Universitas Ma Chung bisa dalam berbagai konteks. Apakah Anda mencari tokoh akademik seperti dosen atau pimpinan, ataukah mahasiswa atau alumni yang berprestasi atau terkenal di bidang tertentu?
3.	Kapan ma chung berdiri	Halo, saya Mei. Saya tidak menemukan informasi di dokumen mengenai kapan Universitas Ma Chung didirikan.	Universitas Ma Chung didirikan pada tanggal 16 April 2007.
4.	apa visi misi ma chung	Halo, saya Mei. Saya tidak menemukan informasi di dokumen mengenai visi dan misi Universitas Ma Chung.	Berdasarkan dokumen yang tersedia, visi Universitas Ma Chung adalah menjadi perguruan tinggi unggul yang berjiwa Pancasila, berwawasan global, serta mengembangkan ilmu pengetahuan dan teknologi untuk kesejahteraan umat manusia. Sementara itu, misi Universitas Ma Chung antara lain menyelenggarakan pendidikan tinggi yang berkualitas, melakukan penelitian dan pengabdian kepada masyarakat, serta membangun kerja sama yang produktif dengan berbagai pihak. Apakah ada bagian visi atau misi yang ingin Anda pahami lebih lanjut?

No.	Pertanyaan	Jawaban Asisten Virtual	Jawaban Gemini
5.	ada fakultas apa saja di binus	Halo, saya Mei. Pertanyaan Anda mengenai fakultas di Binus tidak berkaitan dengan Universitas Ma Chung.	Terkait dengan Universitas Ma Chung dan fakultas nya, dokumen yang Anda unggah tidak memberikan informasi detail tentang fakultas. Namun, pada bagian struktur organisasi, dokumen menyebutkan adanya BPMF di tingkat fakultas. Apakah ada fakultas tertentu yang ingin Anda ketahui?
6.	$5 + 5$ berapa	Halo, saya Mei. Pertanyaan Anda mengenai hasil dari lima ditambah lima tidak berkaitan dengan Universitas Ma Chung.	$5 + 5$ sama dengan 10.
7.	saya harus membayar denda di ma chung 1000 perhari dan saya kena 50 hari, berapa total yang harus saya bayarkan	Halo, saya Mei. Saya tidak menemukan informasi di dokumen yang dapat digunakan untuk menghitung total denda yang harus Anda bayarkan.	Jika denda Anda 1.000 per hari dan Anda terkena denda selama 50 hari, maka total yang harus Anda bayarkan adalah 1.000 dikali 50, yaitu 50.000.
8.	apa itu html	Halo, saya Mei. Pertanyaan Anda mengenai HTML tidak berkaitan dengan Universitas Ma Chung.	HTML adalah singkatan dari HyperText Markup Language. Itu adalah bahasa yang digunakan untuk membuat dan menyusun struktur halaman web. Dengan HTML, Anda bisa menambahkan teks, gambar, tautan, dan elemen lainnya ke dalam halaman web.

Dari hasil jawaban diatas dapat terlihat kemampuan personalisasi asisten virtual. Asisten virtual yang sudah diatur untuk hanya spesifik menjawab pertanyaan yang berkaitan dengan Ma Chung dapat melakukan penolakan pertanyaan yang tidak relevan atau tidak ada pada dokumen yang sudah dipasang. Hal ini berbeda dengan Gemini yang tetap menjawab semua pertanyaan bahkan berhalusinasi dengan memberikan jawaban yang tidak ada pada dokumen. Dari sini dapat dikatakan jika asisten virtual yang dikembangkan memiliki daya personalisasi yang dapat menjadi pembeda utama dari asisten virtual generik seperti Gemini.

4.16 Solusi Keterbatasan TTS Sementara

Karena setelah dilakukan implementasi, F5_TTS memiliki terbatas akan kemampuannya untuk menghasilkan suara dalam bahasa Indonesia, maka pada penelitian ini dilakukan penelitian untuk solusi keterbatasan F5_TTS tersebut. Setelah dilakukan penelitian ditemukan TTS *open source* yang memiliki kemampuan Multilingual yaitu Chatterbox. Setelah dilakukan pengujian *generate* suara dan dengan kapasitas *code* yang masih dapat dimodifikasi untuk memiliki *endpoints*, Chatterbox berhasil menghasilkan suara dalam bahasa Indonesia yang natural. Namun pada waktu diimplementasikan ke dalam sistem, ternyata Chatterbox memiliki keterbatasan berupa latensi yang jauh lebih lama dibanding F5_TTS sekitar 5 detik lebih lama untuk setiap *chunk* dan ketahanan akan *error* yang lemah sehingga ketika diterapkan *chunking* ada beberapa *chunk* yang hilang suaranya.

Untuk alternatif lainnya adalah menggunakan TTS eksternal yaitu ElevenLab. Cara untuk mengintegrasikannya adalah dengan memanfaatkan API yang disediakan, lalu teks yang ingin dihasilkan dikirim dari Web *Application* dan setelah seluruh teks sudah diubah menjadi suara maka akan dikembalikan *response* berupa mp3 yang akan di *play* langsung pada web *application*. Hasil suara yang dihasilkan cukup bagus namun dari sisi latensi masih lebih cepat F5_TTS yang dapat menerapkan sistem *chunking*. Kelemahan berikutnya adalah keterbatasan token berbayar. Dalam penelitian ini diperoleh hasil 366 kata dengan total 2.843 karakter menghabiskan token sekitar 3.000 hingga 3.500 token. Sedangkan dengan menggunakan sampel jawaban pada sesi 2-2, total kata jawaban asisten virtual mencapai 942 kata dengan 4.873 karakter. Pada penawaran ElevenLab berbayar yang paling murah menyediakan penawaran \$5 atau setara sekitar Rp 84.000 hanya memberikan 30.000 token. Jika dilakukan perhitungan kasar maka dengan biaya Rp 84.000 hanya dapat menjawab kurang lebih 30 pertanyaan saja. Data penawaran tersebut tercatat pada bulan Januari 2026.

The screenshot shows the API Pricing page for ElevenLab. It displays several pricing plans:

Plan	Price	Features
Free	\$0 per month	Build for free
Starter	\$5 per month	Choose Starter
Creator	\$11 per month	Choose Creator
Pro	\$99 per month	Choose Pro
Scale	\$330 per month	For Businesses
Business	\$1,320 per month	Business
Enterprise	Custom pricing	Enterprise

Each plan includes a list of features:

- Free:** Text to Speech API, Speech to Text API, Voice Isolator API, Voice Changer API, Dubbing API, Agents, API access.
- Starter:** Everything in Free, plus: Commercial license, Instant Voice Cloning, Dubbing Studio API, Eleven Music API.
- Creator:** Everything in Starter, plus: Professional Voice Cloning, Usage based billing for additional credits, Higher quality audio 192 kbps.
- Pro:** Everything in Creator, plus: 44.1kHz PCM audio output via API.
- Scale:** 30k credits per month.
- Business:** 100k credits per month.
- Enterprise:** 500k credits per month.

Gambar 4.50 Harga Penawaran ElevenLab

4.17 Github Repository

Hasil luaran dari penelitian ini juga berupa sebuah proyek yang dipublikasikan pada platform GitHub, yang berfungsi sebagai salah satu bentuk *remote source code control*. Proyek ini menyimpan seluruh kode sumber hasil penelitian secara terstruktur dan terdokumentasi, sehingga memungkinkan pengelolaan versi, kolaborasi, serta pelacakan perubahan kode secara sistematis. Dengan menggunakan GitHub, proyek dapat dengan mudah di-*clone* atau direplikasi di lingkungan pengembangan lain, baik untuk keperluan pengujian, pengembangan lanjutan, maupun replikasi hasil penelitian, sehingga mendukung prinsip keterbukaan, keberlanjutan, dan reproduktibilitas penelitian. Untuk mengaksesnya dapat dilakukan pada <https://github.com/MiBlue027/Virtual-Assistant/tree/main>.

BAB V

Kesimpulan dan Saran

5.1 Kesimpulan

Penelitian ini berhasil mengembangkan asisten virtual berbasis suara menggunakan Generative AI LLM dengan arsitektur microservice untuk layanan akademik Universitas Ma Chung. Sistem ini mengintegrasikan STT browser API, F5-TTS dengan optimasi chunking dan streaming, RAG berbasis Supabase vector database, serta Gemini 2.5 Flash melalui orkestrasi n8n di Docker. Evaluasi terhadap 30 responden menunjukkan hasil yang baik dengan nilai SUS 77,92, Angket Kepuasan 4,21 dari 5, MOS 3,72 dari 5, dan WER 13,13%, yang menandakan sistem berjalan stabil dan memenuhi target yang sudah ditetapkan pada rumusan masalah. Hasil perbandingan dengan Gemini juga menunjukkan keunggulan asisten virtual dari sisi akurasi, dengan total skor akurasi 86,25% dibandingkan 43,12% pada Gemini, yang membuktikan efektivitas penerapan RAG dan pengelolaan *knowledge base* terkontrol. Dari sisi latensi, asisten virtual memiliki rata-rata 8,24 detik, lebih tinggi dibandingkan Gemini dengan 5,1 detik, namun hal ini dapat dimaklumi karena perbedaan spesifikasi infrastruktur. Selain itu, asisten virtual mampu melakukan personalisasi dengan menolak pertanyaan yang tidak relevan atau tidak tersedia dalam dokumen, berbeda dengan Gemini yang tetap menjawab dan berpotensi berhalusinasi, sehingga sistem ini memiliki nilai tambah sebagai asisten virtual yang terarah dan sesuai kebutuhan institusi pendidikan.

5.2 Saran

Peningkatan kualitas *output* suara merupakan fitur utama yang sangat perlu dikembangkan lebih lanjut. Disarankan agar implementasi TTS berikutnya dapat menghasilkan suara dalam bahasa Indonesia secara baik, bukan suara logat bahasa Inggris yang dipaksakan menjadi bahasa Indonesia. Hal ini sangat penting karena hampir semua pengguna di Universitas Ma Chung menggunakan bahasa Indonesia. Berikutnya juga optimalisasi fitur STT untuk lebih meminimalkan *Word Error Rate* (WER), khususnya pada kata "Ma Chung". Tidak berhenti pada STT nya saja, tetapi kemampuan *Generative AI* harus lebih tahan terhadap *typo* atau deviasi kata yang bisa terjadi karena kegagalan sistem STT.

Selain peningkatan fungsionalitas inti, peningkatan pengalaman pengguna juga dapat menjadi fokus pengembangan berikutnya. Untuk bertransisi dari interface berbasis teks murni, dapat ditingkatkan dengan *mengintegrasikan real-time character animation*. Ini mungkin dapat dikembangkan dengan menggunakan *library grafis high-performance* seperti Three.js untuk renderisasi avatar 3D yang dinamis. Karakter virtual ini tidak hanya berfungsi sebagai elemen visual, tetapi juga sebagai representasi entitas komunikatif yang mampu menunjukkan ekspresi dan gestur tersinkronisasi, sehingga meningkatkan pengalaman pengguna dan lebih menarik untuk dipromosikan.



UNIVERSITAS
MA CHUNG

Daftar Pustaka

- Adiputra, F. (2015). *Container dan Docker: Teknik Virtualisasi dalam Pengelolaan Banyak Aplikasi Web*. Simantec: Jurnal Sistem Informasi dan Teknologi, 4(3). <https://doi.org/10.21107/simantec.v4i3.1384>
- Akhilesh Gadde. (2025). Democratizing Software Engineering through Generative AI and Vibe Coding: The Evolution of No-Code Development. *Journal of Computer Science and Technology Studies*, 7(4), 556-572. <https://doi.org/10.32996/jcsts.2025.7.4.66>
- Amrullah, A. Z., Anas, A. S., & Primajati, G. (2022). *Implementasi Chatbot sebagai Virtual Assistant Penerimaan Mahasiswa Baru pada Universitas Bumigora*. *Jurnal Bumigora Information Technology (BITe)*, 4(1), 17-26. <https://doi.org/10.30812/bite.v4i1.1664>
- Kambau A R. (2024). *Proses Transformasi Digital Pada Perguruan Tinggi Di Indonesia*. *Jurnal Rekayasa Sistem Informasi Dan Teknologi*, 1(3), 126–136. <https://doi.org/10.59407/jrsit.v1i3.481>
- Chen Y, Esmaeilzadeh P. (2024). *Generative AI in Medical Practice: In-Depth Exploration of Privacy and Security Challenges*. *J Med Internet Res* 2024;26:e53008. DOI: <https://doi.org/10.2196/53008>
- Chen, Y., Ju, Z., Xin, X., Xue, L., Li, Y., Jin, Z., Ren, Y., Tian, Q., & Jin, X. (2024). *F5-TTS: A Fairytaler that Fakes Fluent and Faithful Speech with Flow Matching*. arXiv preprint [arXiv:2410.06885](https://arxiv.org/abs/2410.06885). <https://arxiv.org/abs/2410.06885>
- Cong-Lem, N., Soyoof, A., & Tseng, D. (2025). A Systematic Review of the Limitations and Associated Opportunities of ChatGPT. *International Journal of Human–Computer Interaction*, 41(7), 3851–3866. <https://doi.org/10.1080/10447318.2024.2344142>
- Coheur, L. (2020). From Eliza to Siri and Beyond. Dalam M. J. Lesot et al. (Eds.), *Information Processing and Management of Uncertainty in Knowledge-Based Systems: 18th International Conference, IPMU 2020, Lisbon, Portugal, June 15–19, 2020, Proceedings, Part I* (Vol. 1237, hlm. 29–41). Springer International Publishing. https://doi.org/10.1007/978-3-030-50146-4_3
- Dyayu L. A., Beny, B., & Yani, H. (2023). *Evaluasi Usability Aplikasi PeduliLindungi Menggunakan Metode Usability Testing dan System Usability Scale (SUS)*. *Jurnal Manajemen Teknologi Dan Sistem Informasi (JMS)*, 3(1), 395–404. <https://doi.org/10.33998/jms.2023.3.1.720>
- Faqihuddin B. S., Taryana A., Zulfa M. I. (2025). *Pengembangan Application Programming Interface (API) Aplikasi Pendekripsi Penyakit Tanaman*

Menggunakan Metode DevOps. Jurnal SINTA: Sistem Informasi dan Teknologi Komputasi, 2(2), 66-75. <https://doi.org/10.61124/sinta.v2i2.42>

Galatang D. H., Suyanto. (2020). *Syllable-Based Indonesian Automatic Speech Recognition*. International Journal on Electrical Engineering and Informatics - Volume 12, Number 4. <https://www.ijeei.org/docs-15732688485fe84298a8595.pdf>

Google. (2025). *Gemini models*. <https://ai.google.dev/gemini-api/docs/models>

Gunutoro, G., Costaner, L., & Lisnawita, L. (2020). *Aplikasi Chatbot untuk Layanan Informasi dan Akademik Kampus Berbasis Artificial Intelligence Markup Language (AIML)*. Digital Zone: Jurnal Teknologi Informasi dan Komunikasi, 11(2), 291-300. <https://doi.org/10.31849/digitalzone.v11i2.5049>

Irawan D., Fauzi. *Implementasi Arsitektur Microservices pada Pengembangan Aplikasi Absensi Web Terdistribusi*, bit-Tech, vol. 7, no. 3, pp. 1118–1127, Apr. 2025.

Jaber K. M., Laf M., Alkatib A. A., AbedAlghafer A. K., Jawad M. A., Ahmad A. Q. (2024). *A Comparative Study for Virtual Personal Assistants (VPA) and State-of-the-Art Speech Recognition Technology*. International Journal of Computational and Experimental Science and Engineering (Vol. 10-No.3, pp. 427-433). [10.22399/ijcesen.383](https://doi.org/10.22399/ijcesen.383)

Kurniadi D., Nuraeni F., Raharja I. T., Mulyani A. (2022). *Perancangan Aplikasi Text To Speech Dalam Bahasa Indonesia Menggunakan Firebase Machine Learning Kit Berbasis Android*. Jurnal Teknologi Informasi Dan Ilmu Komputer, 9(6), 1281-1288. <https://doi.org/10.25126/jtiik.2022965985>

Kusmayadi, Kurniadi D., Setiawan R. (2023). *Aplikasi Voice Assistant Pada Smartwatch Menggunakan Open Artificial Intelligence*. Journal Algoritma. DOI: [10.33364/algoritma/v.21-2.1499](https://doi.org/10.33364/algoritma/v.21-2.1499)

Lewis P., Perez E., Piktus A., Petroni F., Karpukhin V., Goyal N., Kuttler H., Lewis M., Yih W., Rocktäschel T., Riedel S., Kiela D. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. Arxiv. <https://doi.org/10.48550/arXiv.2005.11401>

Ma Chung. (2025). *Dokumen Penting*. <https://machung.ac.id/dokumen-penting/>

Maola, P. S., Karai Handak, I. S., & Herlambang, Y. T. (2024). *Penerapan Artificial Intelligence Dalam Pendidikan Di Era Revolusi Industri 4.0*. Educatio, 19(1), 61–72. <https://doi.org/10.29408/edc.v19i1.24772>

Microsoft. (2024). AI at Work Is Here. Now Comes the Hard Part. <https://www.microsoft.com/en-us/worklab/work-trend-index/ai-at-work-is-here-now-comes-the-hard-part>

Mulyatun S., Utama H., Mustopa AA. (2021). *Pendekatan Natural Language Processing Pada Aplikasi Chatbot Sebagai Alat Bantu Customer Service*. Journal of Information System Management (JOISM), 2(2), 12-17. <https://doi.org/10.24076/joism.2021v3i1.404>

Novianto, M. A., & Munir, S. (2022). *Analisis dan Implementasi Restful API guna Pengembangan Sistem Informasi Akademik pada Perguruan Tinggi*. Jurnal Informatika Terpadu, 8(1), 47–61. <https://doi.org/10.54914/jit.v8i1.409>

Octavia, A. C. B., & Nugraha, J. (2024). *Influence of Enjoyment and Trust on the Use of Artificial Intelligence-based Voice Assistant in Vocational Students Using Technology Acceptance Model (TAM)*. Journal of Office Administration : Education and Practice, 4(1), 10–23. <https://doi.org/10.26740/joaep.v4n1.p10-23>

Okpomu, Bethel E. (2024). *Security and Privacy Concerns in The Implementation of Virtual Assistants in Multinational Companies*. Federal Polytechnic, Ekowe, Bayelsa State,Nigeria (Val. 1, 17-31). American Journal Of Current Tendency and Innovation. <https://publishingjournals.org/ajcti/article/download/42/36/105>

OpenAi. (2023). GPT-4 Technical Report. <https://doi.org/10.48550/arXiv.2303.08774>

Petrus J., Eematita, Sukemi, Erwin. (2023). *A Novel Approach: Tokenization Framework based on Sentence Structure in Indonesian Language*. International Journal of Advanced Computer Science and Applications, Vol. 14, No. 2, 2023. https://thesai.org/Downloads/Volume14No2/Paper_64-A_Novel_Approach_Tokenization_Framework.pdf

Perdana, R. P., & Irwansyah, I. (2019). *Implementasi Asisten Virtual Dalam Komunikasi Pelayanan Pelanggan (Studi Kasus Pada Layanan Pelanggan Telkomsel)*. Jurnal Komunikasi, 11(2), 183–196. <https://doi.org/10.24912/jk.v11i2.5491>

Prabowo, M., & Suprapto, A. (2021). *Usability Testing pada Sistem Informasi Akademik IAIN Salatiga Menggunakan Metode System Usability Scale*. JISKA (Jurnal Informatika Sunan Kalijaga), 6(1), 38–49. <https://doi.org/10.14421/jiska.2021.61-05>

Pratama I. I. R., Sisepaputra B. (2024). *Pengembangan Sistem Helpdesk Menggunakan Chatbot Dengan Metode Retrieval-augmented Generation (Rag)*. Journal of Informatics and Computer Science (JINACS). Vol. 6 No. 03. <https://doi.org/10.26740/jinacs.v6n03.p696-710>

- Rahmawati, L., & Sumarsono, S. (2024). *Desain Pengembangan Website dengan Arsitektur Model View Controller pada Framework Laravel*. Jurnal Teknologi Dan Sistem Informasi Bisnis, 6(4), 785-790. <https://doi.org/10.47233/jteksis.v6i4.1497>
- Salamun, S., Aldi Aprialdo, & Sukri. (2024). *Optimasi Chatbot dengan Pemanfaatan Natural Language Processing*. Jurnal Komputer Terapan, 10(1), 17–26. <https://doi.org/10.35143/jkt.v10i1.6181>
- Salsabila, S., & Sohidin, S. (2024). *Pemahaman Etika Akademik Mahasiswa dalam Penggunaan Artificial Intelligence (AI)*. Journal of Education Research, 5(4), 6671–6680. <https://doi.org/10.37985/jer.v5i4.1944>
- Samudra, G., Zy, A. T., & Ermanto. (2025). *Implementation of Retrieval Augmented Generation (RAG) in the Design of Digestive Health Chatbot*. Journal of System and Artificial Intelligence (JSAI), 8(1). <https://doi.org/10.36085/jsai.v8i1.7678>
- Sinlae, F., Irwanda, E., Maulana, Z., & Eka Syahputra, V. (2024). *Penggunaan Framework Laravel dalam Membangun Aplikasi Website Berbasis PHP*. Jurnal Siber Multi Disiplin, 2(2), 119–132. <https://doi.org/10.38035/jsmd.v2i2.186>
- Tan J. T., Gan R. K., Alsua C., Peterson M., Sales R. U., Gan A. Z., Martinez J. A. C., Gonzalez P. A. (2025). *Psychological First Aid by AI: Proof-of-Concept and Comparative Performance of ChatGPT-4 and Gemini in Different Disaster Scenarios*. Journal of Clinical Psychology. Volume81, Issue8. <https://doi.org/10.1002/jclp.23808>
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). *Attention Is All You Need*. Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA. <https://doi.org/10.48550/arXiv.1706.03762>
- Wang X., Li H., Zhang Z., Chen H., Zhi W. (2025). *Modular Machine Learning: An Indispensable Path towards New-Generation Large Language Models*. <https://arxiv.org/html/2504.20020v1>
- Weber, M., Engert, M., Schaffer, N. et al. (2022). *Organizational Capabilities for AI Implementation—Coping with Inscrutability and Data Dependency in AI*. Inf Syst Front 25, 1549–1569 (2023). <https://doi.org/10.1007/s10796-022-10297-y>
- Wessel, M., Adam, M., Benlian, A., Majchrzak, A., & Thies, F. (2025). *Generative AI and its Transformative Value for Digital Platforms*. Journal of Management Information Systems, 42(2), 346–369. <https://doi.org/10.1080/07421222.2025.2487315>

Widiastuti W., Rohayanah, Purnamaningsih I. R. (2025). *Manajemen Mutu Pendidikan Berbasis Digital: Pengaruh Implementasi Sistem Informasi Pendidikan Terhadap Kinerja Akademik Di Perguruan Tinggi*. Journal Development, 13(1), 161–173. <https://doi.org/10.53978/jd.v13i1.572>

Zahro D. A., Nugraha J. (2021). *Analysis Of The Use Of Mendeley Using The Technology Acceptance Model (Tam) Approach: Study In Students Of Office Administration Education In Surabaya State University*. Technology Acceptance Model Vol12, No. 1. DOI: 10.56327/jurnaltam.v12i1.1014



UNIVERSITAS
MA CHUNG

Lampiran

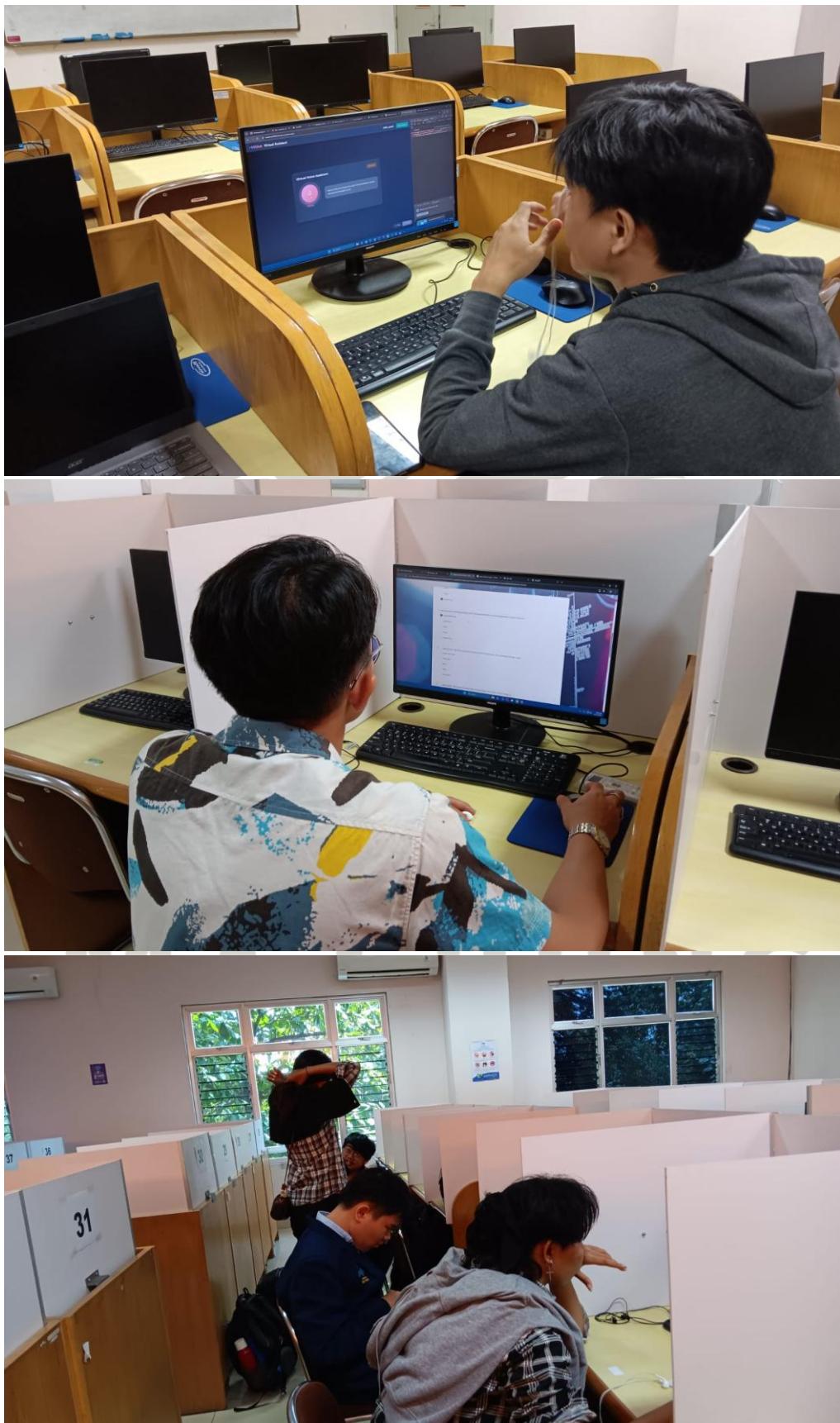
A. Daftar Nama Responden

No.	Nama
1.	Olfat Harits Alatas
2.	Shelly
3.	Benedict Chilwin Punarto
4.	Clint Joy P.M.M
5.	Hessel Putra Kencana
6.	Vannes Yanuarta Salim
7.	Kevin Cheng
8.	Steven Ferdinand
9.	Kinanti Rizki Anggita Putri
10.	Yudhistira Nalendra A A
11.	Adam Chesta
12.	Ivan Lie
13.	Rivian
14.	Annisa Naura Salsabila
15.	Gilbetch Ronaldo Triswanto
16.	Benedict Michael Pepper
17.	Alfond manuel joseph
18.	Elroi Yonatan Raharjo
19.	Sutri Ajeng Neng Rahayu
20.	Cecilia Margaretha
21.	Djoel Gemah
22.	Nathanael
23.	Benedictus Marcello Nathanael
24.	Jeremy Marcellino Limantoro
25.	Reymon Okto Christiaan
26.	Abednego Indra Saputra
27.	Albert William Saputra
28.	Yoachim Yeremia Simatupang
29.	Kent Carol Erikson
30.	Anak Agung Ngurah Rai Ando Pradana

B. Data Respon Mentah

Id	Start time	Completion time	Email	Session Id	Nama	Status
1	12/9/2025 15:40	12/9/2025 15:43	anonymous	2-7e0ea97	Olfat Harits Alatas	Mahasiswa
2	12/9/2025 15:46	12/9/2025 15:48	anonymous	2-1fa8a98	shelly	Mahasiswa
3	12/9/2025 15:54	12/9/2025 15:57	anonymous	2-9e99c64d	Benedict Chilwin Punar	Mahasiswa
4	12/9/2025 16:02	12/9/2025 16:05	anonymous	2-fb3b915d	Clint Joy P.M.M	Mahasiswa
5	12/9/2025 16:14	12/9/2025 16:16	anonymous	2-5e33ee2e	Hessel Putra Kencana	Mahasiswa
6	12/9/2025 16:25	12/9/2025 16:27	anonymous	2-beb70421	Vannes Yanuarta Salim	Mahasiswa
7	12/9/2025 17:11	12/9/2025 17:13	anonymous	2-0c053194	Kevin Cheng	Mahasiswa
8	12/11/2025 9:04	12/11/2025 9:53	anonymous	2-b5d2c576	Steven Ferdinand	Mahasiswa
9	12/11/2025 9:57	12/11/2025 9:59	anonymous	2-8cd12db7	Kinanti Rizki Anggita Pu	Mahasiswa
10	12/11/2025 10:02	12/11/2025 10:03	anonymous	2-af4651ea	Yudhistira Nalendra A	Mahasiswa
11	12/11/2025 10:05	12/11/2025 10:08	anonymous	2-03e0801b	Adam Chesta	Mahasiswa
12	12/11/2025 10:13	12/11/2025 10:17	anonymous	2-2aae19bf	Ivan Lie	Mahasiswa
13	12/11/2025 10:18	12/11/2025 10:21	anonymous	2-19830d6c	rivian	Mahasiswa
14	12/11/2025 10:28	12/11/2025 10:31	anonymous	2-3a35dec8	Annisa Naura Salsabila	Mahasiswa
15	12/11/2025 10:34	12/11/2025 10:37	anonymous	2-8e19e55d	Gilibetch Ronaldo Trisw	Mahasiswa
16	12/11/2025 10:39	12/11/2025 10:42	anonymous	2-97bf06ff	Benedict Michael Pepp	Mahasiswa
17	12/11/2025 10:44	12/11/2025 10:46	anonymous	2-97bf06ff	alfond manuel joseph	Mahasiswa
18	12/11/2025 10:57	12/11/2025 11:01	anonymous	2-1b5fc4b	Elroi Yonatan Raharjo	Mahasiswa
19	12/11/2025 11:01	12/11/2025 11:09	anonymous	2-543f50c5	Sutri Ajeng Neng Rahay	Mahasiswa
20	12/11/2025 11:11	12/11/2025 11:13	anonymous	2-324b96ee	Cecilia Margaretha	Mahasiswa
21	12/11/2025 11:15	12/11/2025 11:17	anonymous	2-8a40336d	Djoel Gemah	Mahasiswa
22	12/11/2025 11:19	12/11/2025 11:21	anonymous	2-03dd3d7d	Nathanael	Mahasiswa
23	12/11/2025 11:24	12/11/2025 11:25	anonymous	2-2344d984	Benedictus Marcello Ni	Mahasiswa
24	12/11/2025 11:28	12/11/2025 11:29	anonymous	2-f2e48bae	Jeremy Marcellino Lim	Mahasiswa
25	12/11/2025 11:35	12/11/2025 11:39	anonymous	2-01135546	Reymon Okto Christia	Mahasiswa
26	12/12/2025 22:14	12/12/2025 22:27	anonymous	2-2c7c1520	Abednego Indra Saputra	Mahasiswa
27	12/12/2025 22:27	12/12/2025 22:30	anonymous	2-bbf3ea48	Albert William Saputra	Mahasiswa
28	12/12/2025 22:30	12/12/2025 22:35	anonymous	2-8cfdd37d	Yoachim Yeremia Simat	Mahasiswa
29	12/12/2025 22:35	12/12/2025 22:39	anonymous	2-5967333e	Kent Carol Erikson	Mahasiswa
30	12/12/2025 22:54	12/12/2025 22:58	anonymous	2-0c81c988	Anak Agung Ngurah Ra	Mahasiswa

C. Dokumentasi Evaluasi Dengan Responden



D. Contoh Potongan Code Controller

```
4
5     use App\Service\AIService\N8nSvc;
6     use Exception;
7
8     8 usages  ↳ MiBlue027 +1
9
10    class VirtualAssistantController
11    {
12        no usages  ↳ MiBlue Project
13        public function VirtualAssistantView()
14        {
15            view( view: "virtual_assistant_view", [
16                "title" => "Assistant Virtual"
17            ]);
18        }
19
20        no usages  ↳ MiBlue027
21        public function VirtualAssistantVoiceView()
22        {
23            view( view: "virtual_assistant_voice_view", [
24                "title" => "Assistant Virtual"
25            ]);
26        }
27
28        no usages  ↳ MiBlue Project
29        public function upload_doc_view()
30        {
31            view( view: "doc_upload_view", [
32                "title" => "upload doc"
33            ]);
34        }
35    }
```

E. Contoh Potongan Code Service

F. Contoh Potongan Code View

```
1 <div class="dUICChatBotContainer">
2   <?php include __DIR__ . '/head_nav.php'; ?>
3
4   <div class="va_voice-particles" id="va_voice-particles"></div>
5
6   <div class="dUICChatBot-MessageBoxContainerJQ dUICChatBot-MessageBoxContainer"></div>
7
8   <div class="dUICChatBotFormContainer">
9     <form id="chatBot" class="dUICChatBotFormJQ dUICChatBotForm" data-url="/ai/n8n/chat-bot">
10       <label for="txaUICChatBot_chatBot"></label>
11       <textarea id="txaUICChatBot_chatBot" placeholder="How can I Help you?" rows="1" class="txaUicChatBotJQ txaUicChatBot"></textarea>
12       <button class="btnUICChatBot-SendMessage" type="submit">
13         
14       </button>
15       <button class="btnUICChatBot-STT-JQ btnUICChatBot-STT" type="button">
16         
22       </button>
23     </form>
24   </div>
25
26 </div>
27
28 <div class="chatMicModeToggleJQ chatMicModeToggle va_voice-mode-toggle">
29   <div class="chatMicModeOption chatMicModeOptionChat active va_voice-mode-option" data-mode="chat">
30     <svg xmlns="http://www.w3.org/2000/svg" width="16" height="16" viewBox="0 0 24 24" fill="none" stroke="currentColor" stroke-width="2">
31       <path d="M21 15z 2 0 0 1-2 2H-1-4 4V5a2 2 0 0 1 2-2H4a2 2 0 0 1 2 2z"/>
32     </svg>
33   </div>
34 </div>
```

G. Contoh Pencatatan Data Mentah Pengujian Perbandingan Peforma Asisten Virtual Dengan Gemini

Evaluasi VA Tukarair - Last Modified: Wed at 01:15												
File Home Insert Draw Page Layout Formulas Data Review View Automate Help												
Aptos Narrow 11 A A Alignment Merge & Center Conditional Formatting Number Styles												
A2	B2	C2	D2	E2	F2	G2	H2	I2	J2	K2	L2	M2
1	Pertanyaan	Response Asisten Virtual	Response Guru	Jawaban Siswa	Latitude VA	Latitude Guru	Kesimpulan	Skor rasa	Skor genial			
1	What is the content of the memo from Vice Chancellor 2	He/Him. I am the memo from Vice Chancellor 2, with reference number: M/2024/0001/0001/0001/0001. Concerning the new admission fees for the Recognition of Prior Learning or RPL pathway for the academic year 2024 or 2025.	Ma/Chung. Ma/Chung has informed me that information regarding the new admission fees for the Recognition of Prior Learning or RPL pathway for the academic year 2024 or 2025.	Ma/Chung. Ma/Chung has informed me that information regarding the new admission fees for the Recognition of Prior Learning or RPL pathway for the academic year 2024 or 2025.	8.23	3.25	Vl lebih baik	2	0			
2	Sebutkan semboyan universitas ma chung	The university motto is as follows: For the Faculty of Economics and Business, the color is red. For the Faculty of Law, the color is orange. For the Faculty of Languages, the color is white. With Contribution, We Attain Success.	Semboyan Universitas Ma Chung adalah "Integrity dan Excellent". Hal ini tercantum dalam dokumen peraturan akademik tersebut.	Sembayang a. with Quality, We Stand; b. with Originality, We Flourish; c. with Win-Win Effect, We Cooperate; d. with Contribution, We Attain Success.	8.86	5.49	Vl lebih baik	2	0			
3	State the color of the faculty flag.	The faculty flag colors are as follows: For the Faculty of Economics and Business, the color is red. For the Faculty of Law, the color is orange. For the Faculty of Languages, the color is white. For the Faculty of Health Sciences, the color is yellow.	Dokumen Peraturan Akademik Universitas Ma Chung 2024 tidak menyebutkan warna bendera fakultas secara detail. Apakah ada hal lain yang ingin diambil?	a. Fakultas Ekonomi dan Bisnis: Merah. b. Fakultas Teknologi dan Desain: Hijau. c. Fakultas Bahasa Puth. d. Fakultas Ilmu Kesehatan: Kuning.	14.21	5.26	Vl lebih baik	2	0			
4	Cantikkan sejarah universitas ma chung	Ma/Chung, Universitas Ma Chung was established on July 7, 2004, and its inauguration took place on March 20, 2007. The university was founded by a family of scholars from the Ma Chung Tongiao community. The initial idea for its establishment came from the late Prof. Dr. H. Ma Chung, who had the vision to establish a university in Malang. This vision was realized when the university was formed in 2004. In 2007, the Ma Chung Education Foundation was established as the managing body of the university. The name Ma Chung Tongiao "Ching" in the Hellenic tradition also "progress" and "prosperity" in Ma/Chung.	Menurut Dokumen Peraturan Akademik, Universitas Ma Chung didirikan pada tahun 2007 di Malang, Jawa Timur. Universitas ini memiliki motto "Integrity dan Excellent" dan berjalinan untuk mengembangkan individu yang berintegritas dan berkarakter. Apakah ada detail lebih lengkap yang ingin Anda tanyakan?	1. Ma/Chung adalah sebuah kota yang berada di Provinsi Jawa Timur. 2. Ma/Chung merupakan sebuah kota yang berada di Provinsi Jawa Timur. China pada masa Sungguan. 3. Dilandanya semangat "hukum, beradil, dan mengabdi kepada masyarakat". 4. PFI Chung didirikan pada 1 Maret 2004. Perpaduan antara Ma Chung dengan PFI Chung. 5. Prof. Dr. H. Ma Chung, SE, MM, M.A., M.Pd., M.Si., Dr. Sugih Kurniati, Heni Sunita, Komaria Lositi, Elvandy Sutrisno, Agus Chandra, Hadi Widjaja, Nurayati Tamansugih, Neneh, dan Lestari. 6. Ma Chung merupakan sebuah kota yang berada di Nagawi/daerah Xian, dan Serbono Wintoro. 8. Pada result ulang pertemuan yang telah ke-60 SMU Ma Chung di Malang, Jawa Timur. 9. Universitas Ma Chung dibentuk pada tahun 2007 di Malang, Jawa Timur. Universitas Ma Chung di Malang pada 17 Juli 2007. 10. Universitas Ma Chung dibentuk dan direncanakan pada 7 Juli 2007. 11. Yayasan Harapan Bangsa Sejahtera (YBS) dibentuk untuk menaungi Universitas.	5.4	3.21	Vl lebih lengkap, Siswa benar topik	2	1			
	Seisi-1	Seisi-2	Seisi-1	Seisi-2	Seisi-3	Seisi-2	Seisi-1	Seisi-2	Irrelevant			

