BAB III METODE PENELITIAN

3.1 Jenis Penelitian

Penelitian ini menggunakan metode *Research and Development* (R&D). Metode R&D merupakan metode penelitian yang digunakan untuk menghasilkan produk tertentu dan menguji efektivitas produk tersebut. Metode ini dipilih karena tujuan utama penelitian ini adalah merancang dan membangun sistem berupa asisten virtual kampus berbasis suara yang mengintegrasikan beberapa teknologi pemrosesan bahasa alami, navigasi, dan interaksi karakter virtual. Pendekatan R&D ini memungkinkan peneliti untuk melakukan pembangunan melalui tahapan-tahapan yang sistematis dan terstruktur dari analisis kebutuhan hingga evaluasi produk akhir.

Ruang lingkup penelitian ini dibatasi pada pengembangan asisten virtual untuk Fakultas Ilmu Komputer (Fasilkom) Universitas Mercu Buana Meruya. Data yang digunakan meliputi data dosen Fasilkom, tenaga kependidikan Fasilkom, jadwal mengajar dosen Fasilkom, dan denah Univesitas Mercu Buana Meruya. Sistem yang dikembangkan ini dirancang untuk menjalankan sebagian besar proses komputasi pada perangkat keras lokal dengan spesifikasi menengah, termasuk inferensi model bahasa, pengenalan entitas, pencarian rute, dan konversi suara karakter, sehingga meminimalkan ketergantungan terhadap layanan komputasi awan (*cloud*). Dan untuk modul *Text-to-Speech* yang dimana modul ini memanfaatkan layanan dari Edge-TTS yang memerlukan koneksi internet untuk melakukan sintesis suara.

3.2 Tahapan Penelitian

Tahapan penelitian ini mengacu pada model pengembangan R&D yang terdiri dari enam tahap utama. Pada fase pengembangan sistem, pendekatan yang digunakan mengadopsi model *Waterfall*, di mana setiap tahap dilaksanakan secara berurutan dan hasil dari satu tahap menjadi masukan bagi tahap berikutnya. Berikut ini adalah penjelasan dari masing-masing tahap penelitian :

1. Analisis Kebutuhan

   Analisis kebutuhan dilakukan melalui observasi langsung peneliti sebagai mahasiswa aktif di Fakultas Ilmu Komputer Universitas Mercu Buana. Berdasarkan pengalaman tersebut, ada beberapa permasalahan utama yang dihadapi oleh mahasiswa baru Fasilkom dan tamu kampus, yaitu kesulitan dalam mencari informasi mengenai dosen atau tenaga kependidikan, keterbatasan mengakses jadwal dosen secara *real-time*, serta minimnya panduan navigasi untuk menemukan lokasi ruangan di dalam lingkungan kampus. Berdasarkan hasil analisis, ditetapkan kebutuhan fungsional sistem yang mencakup kemampuan menerima input suara, mengenali entitas kampus, menjawab pertanyaan jadwal, serta menyediakan navigasi rute pada denah kampus.

2. Pengumpulan Data

   Tahap ini meliputi pengumpulan dan penyusunan data untuk *knowledge base* yang menjadi sumber informasi bagi asisten virtual kampus. Data yang dikumpulkan terdiri dari data profil dosen dan tenaga kependidikan Fasilkom, data jadwal mengajar dosen Fasilkom, serta gambar denah Universitas Mercu Buana Meruya.

3. Perancangan Sistem

   Pada tahap perancangan ini meliputi arsitektur keseluruhan sistem, yang mencakup alur pemrosesan input suara hingga output respons, desain *knowledge base*, mekanisme pengenalan entitas, dan navigasi kampus.

4. Pengembangan dan Implementasi

   Tahap ini merupakan realisasi rancangan ke dalam kode program menggunakan bahasa pemrograman Python dengan *framework* Flask. Antarmuka pengguna dibangun menggunakan HTML, CSS, dan JavaScript dengan integrasi Live2D.

5. Pengujian

   Pengujian dilakukan dengan menggunakan metode pengujian *black box* melalui serangkaian skenario percakapan yang mencakup pertanyaan informasi dosen, jadwal, lokasi, navigasi, serta pertanyaan yang diproses oleh LLM. Aspek yang diuji meliputi akurasi pengenalan entitas oleh

modul NER, ketepatan klasifikasi intent, kebenaran rute yang dihasilkan oleh algoritma A*, serta responsivitas keseluruhan sistem. Setiap skenario didokumentasikan dengan mencatat *input*, *output* yang diharapkan.

6. Evaluasi

   Tahap akhir ini menganalisis hasil pengujian secara keseluruhan untuk menilai sejauh mana sistem memenuhi tujuan penelitian. Evaluasi mencakup analisis tingkat keberhasilan setiap komponen, identifikasi kekuatan dan kelemahan sistem, serta saran untuk pengembangan selanjutnya.

3.3 Desain Penelitian

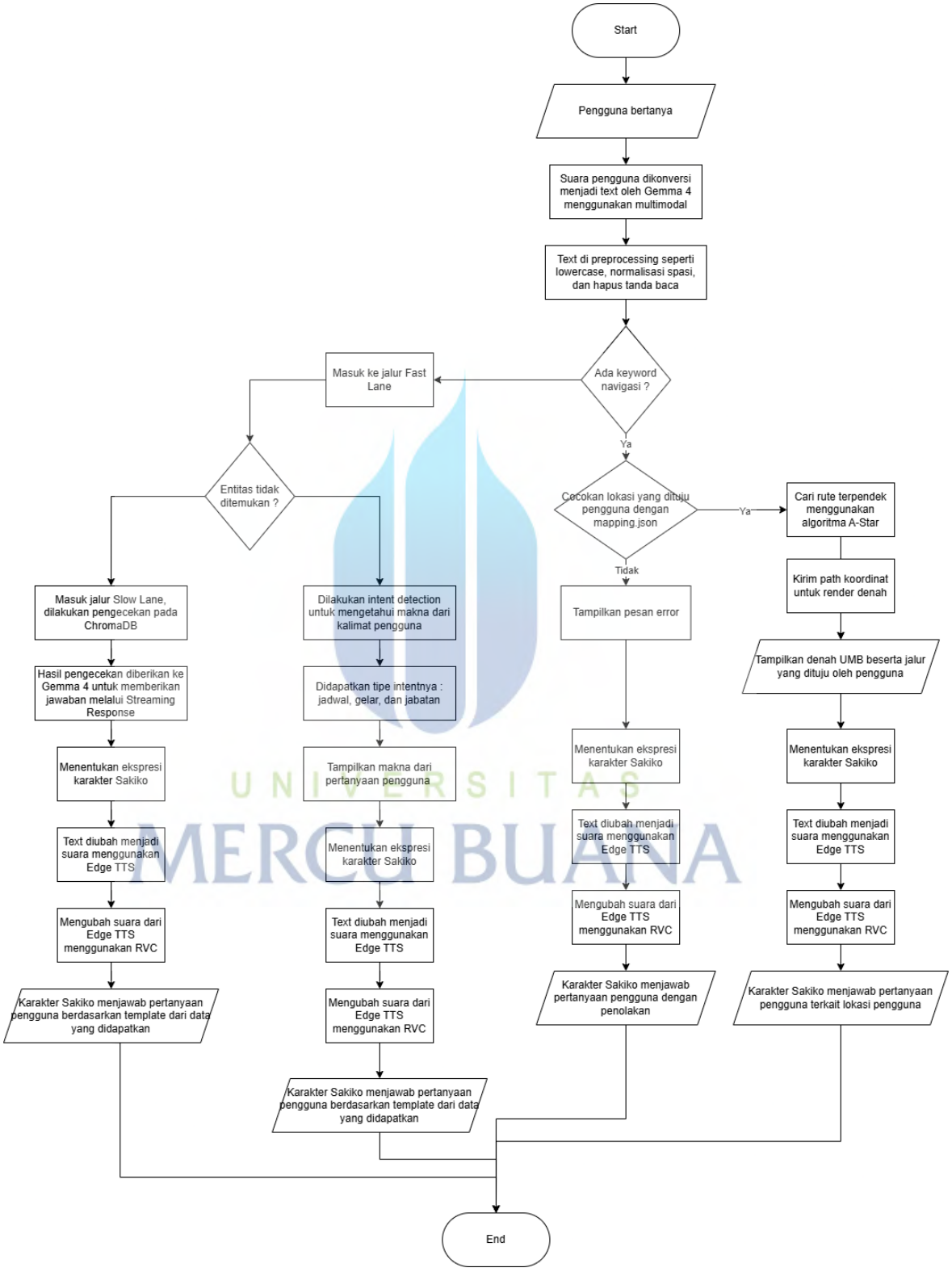
Desain penelitian ini mencakup perancangan arsitektur sistem secara keseluruhan, struktur basis pengetahuan, mekanisme pengenalan entitas, serta mekanisme navigasi kampus. Setiap komponen dirancang untuk saling terintegrasi dalam satu alur pemrosesan yang terpadu, mulai dari penerimaan *input* suara pengguna hingga pengiriman respons dalam bentuk teks dan audio.

3.3.1 Arsitektur Sistem

Arsitektur sistem asisten virtual kampus dirancang menggunakan model *client server* dengan komunikasi *real time* melalui protokol *Server Sent Events* (SSE). Sisi client berupa aplikasi web yang berjalan pada browser, terdiri dari antarmuka karakter Live2D, area percakapan, dan kanvas peta navigasi. Dari sisi server berupa *framework* Flask yang mengatur seluruh alur pemrosesan, mulai dari penerimaan audio hingga pengiriman respons dalam bentuk stream audio dan teks.

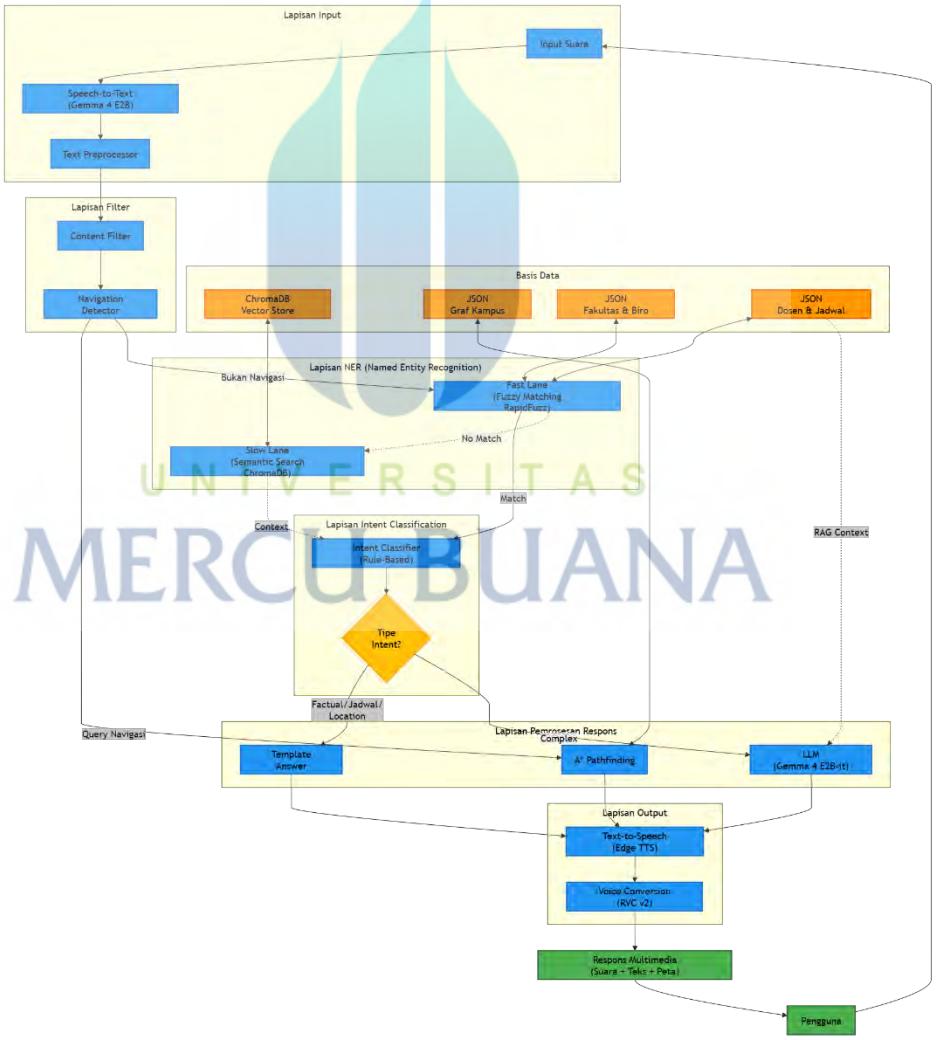
Server Flask berkomunikasi dengan llama-server yang menjalankan model Gemma 4 E2B secara lokal pada port 8080. Llama-server dijalankan menggunakan llama.cpp dengan parameter GPU offloading (-ngl 10) untuk mengakselerasi inferensi melalui CUDA. Kedua server dijalankan secara bersamaan melalui script start.bat yang mengatur urutan inisialisasi: llama-server terlebih dahulu (dengan jeda 8 detik untuk loading model), kemudian

Flask server, dan terakhir membuka browser secara otomatis. Berikut flowchart dari asisten virtual kampus :



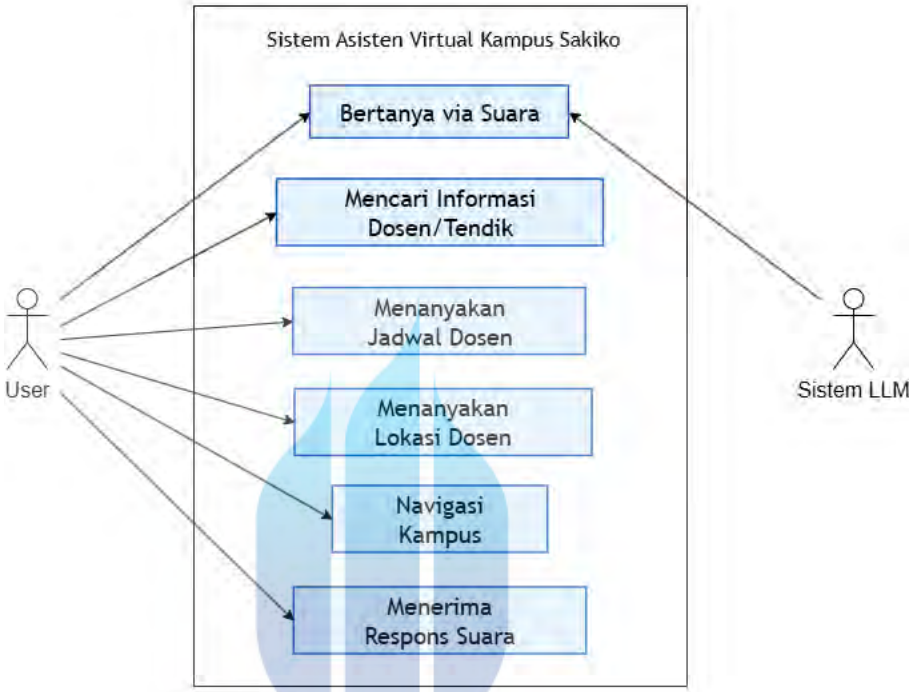
Gambar 3.1 Flowchart asisten virtual kampus

Arsitektur sistem menggambarkan alur integrasi seluruh modul utama dalam mengolah interaksi pengguna secara *end-to-end*. Alur kerja dimulai dari modul *Speech-to-Text* (STT) untuk konversi audio pengguna, dilanjutkan dengan modul *Text Preprocessor* dan *Content Filter*. Pemrosesan informasi dilakukan melalui *Hybrid NER* dan *Intent Classifier* untuk menentukan apakah respons diambil dari basis data pengetahuan, algoritma navigasi A*, atau *Large Language Model* (LLM). Hasil jawaban teks kemudian disintesis menjadi keluaran audio melalui modul *Text-to-Speech* (TTS) dan *Voice Conversion* (RVC v2). Gambaran lengkap alur integrasi komponen tersebut ditunjukkan pada Gambar 3.2 aristektur sistem asisten virtual kampus.



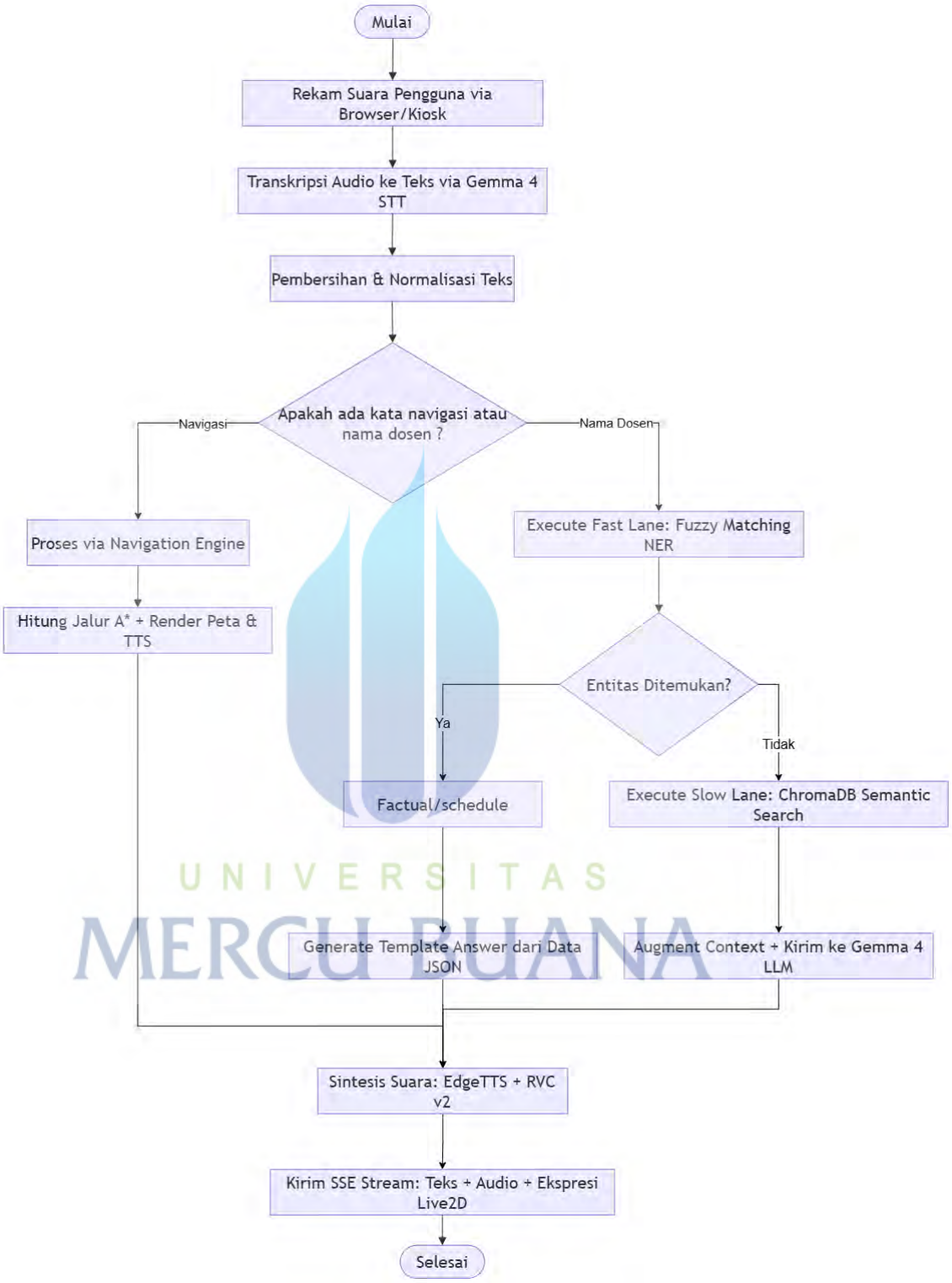
Gambar 3.2 Arsitektur Sistem

Untuk menggambarkan perancangan fungsionalitas sistem serta bentuk interaksi antara pengguna dengan asisten virtual Sakiko, disusun *Use Case Diagram*. Diagram ini memetakan hubungan antara aktor utama dengan seluruh fitur utama yang disediakan oleh sistem sebagaimana ditunjukkan pada gambar 3.3 *use case diagram*



Gambar 3.3 *Use Case Diagram*

Berdasarkan gambar 3.3 di atas, sistem melibatkan dua aktor utama, yaitu Mahasiswa yang berinteraksi secara lisan melalui input suara untuk mengakses informasi dosen, jadwal, dan navigasi kampus. Selain itu, *LLM Server* bertindak sebagai aktor sekunder yang memproses transkripsi suara dan generasi respons sistem secara dinamis.



Gambar 3.4 *Use Case Diagram*

Berdasarkan Gambar 3.4, alur kerja sistem diawali dari pengolahan input suara pengguna menjadi teks hingga tahap penyaringan konten. Selanjutnya, kueri diproses berdasarkan jenisnya: permintaan rute diproses

menggunakan algoritma A*, pertanyaan faktual/jadwal dijawab secara langsung melalui *template*, sedangkan kueri kompleks diteruskan ke LLM Gemma 4 dengan bantuan pencarian semantik ChromaDB. Pada tahap akhir, seluruh respons teks dikonversi menjadi suara karakter Sakiko (Edge TTS + RVC v2) dan ditampilkan kembali kepada pengguna

3.3.2 Desain Knowledge Base

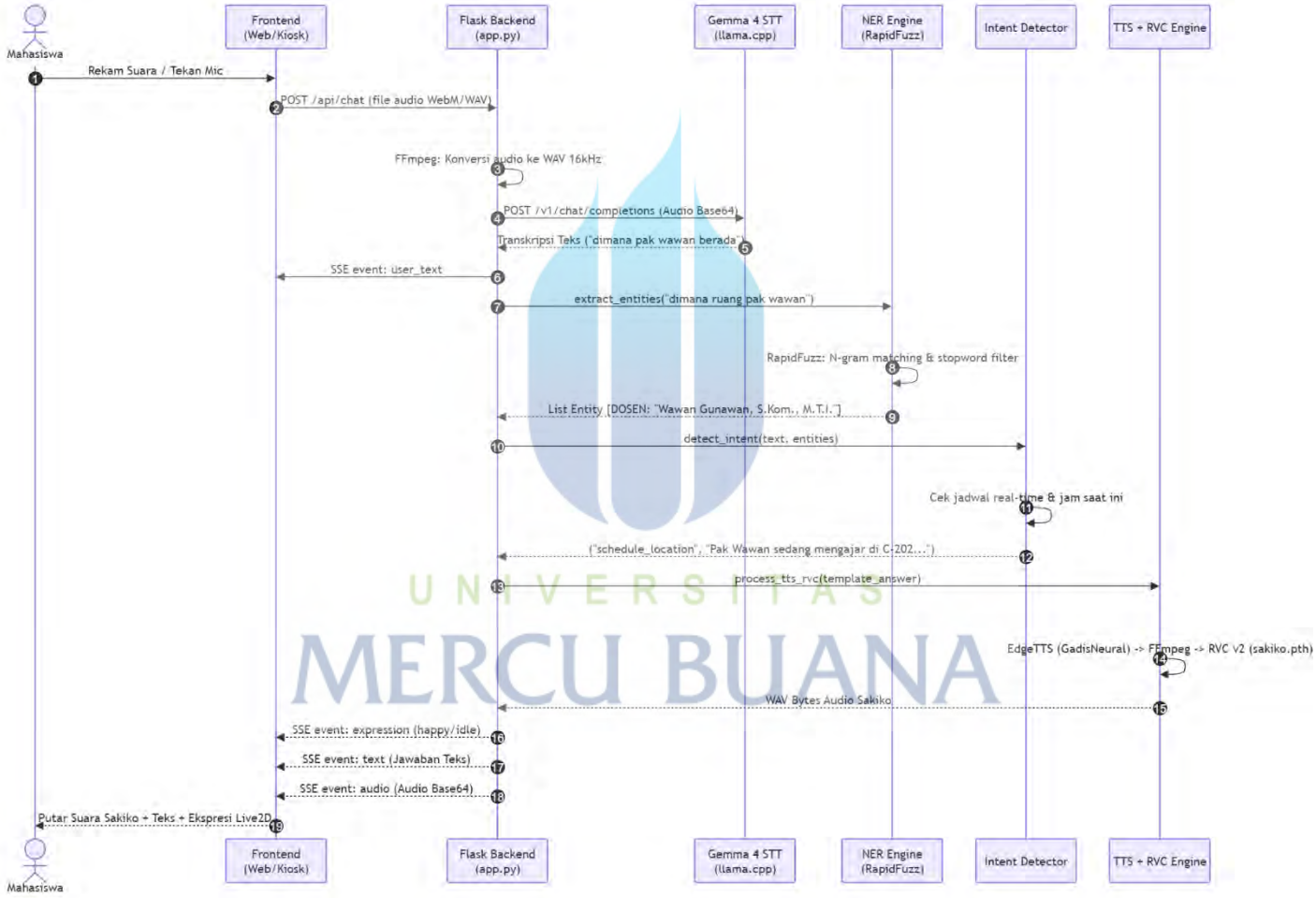
Knowledge base sistem disimpan dalam format JSON yang terdiri dari beberapa file terpisah berdasarkan kategori data. File *dosen_tendik_fasilkom.json* menyimpan data profil dosen dan tenaga kependidikan yang mencakup nama lengkap dengan gelar, jabatan, kategori, bidang ilmu, riwayat pendidikan (S1, S2, S3), serta daftar alias untuk pencocokan NER. File *jadwal_dosen_fasilkom.json* menyimpan data jadwal dosen Fasilkom mengajar yang mencakup hari, jam mulai, jam selesai, mata kuliah, ruang kelas, gedung, serta referensi node navigasi. Kedua file ini digabungkan saat inisialisasi sistem berdasarkan *field* nama untuk menghasilkan satu data terpadu per dosen.

Untuk keperluan navigasi, terdapat tiga file tambahan: *graph.json* yang mendefinisikan simpul dan sisi graf navigasi kampus, *locations.json* yang menyimpan informasi deskriptif setiap lokasi, serta *mapping.json* yang memetakan kata kunci pencarian ke identifier simpul graf. Keseluruhan data ini membentuk basis pengetahuan yang digunakan oleh modul NER, *intent detector*, dan *navigation engine* untuk menghasilkan respons yang akurat.

3.3.3 Desain Hybrid NER

Hybrid NER dirancang dengan dua jalur pemrosesan yang saling melengkapi. Jalur cepat (*fast lane*) bekerja dengan membuat *n-gram* berukuran satu hingga lima kata dari teks input, mengurutkannya dari yang terpanjang, memfilter *stopword*, kemudian mencocokkan setiap *n-gram* terhadap basis data alias menggunakan algoritma *fuzzy matching* dari library *RapidFuzz*. Sistem menerapkan *dynamic threshold* yang menyesuaikan ambang batas kemiripan berdasarkan panjang teks: teks sangat pendek (1-3

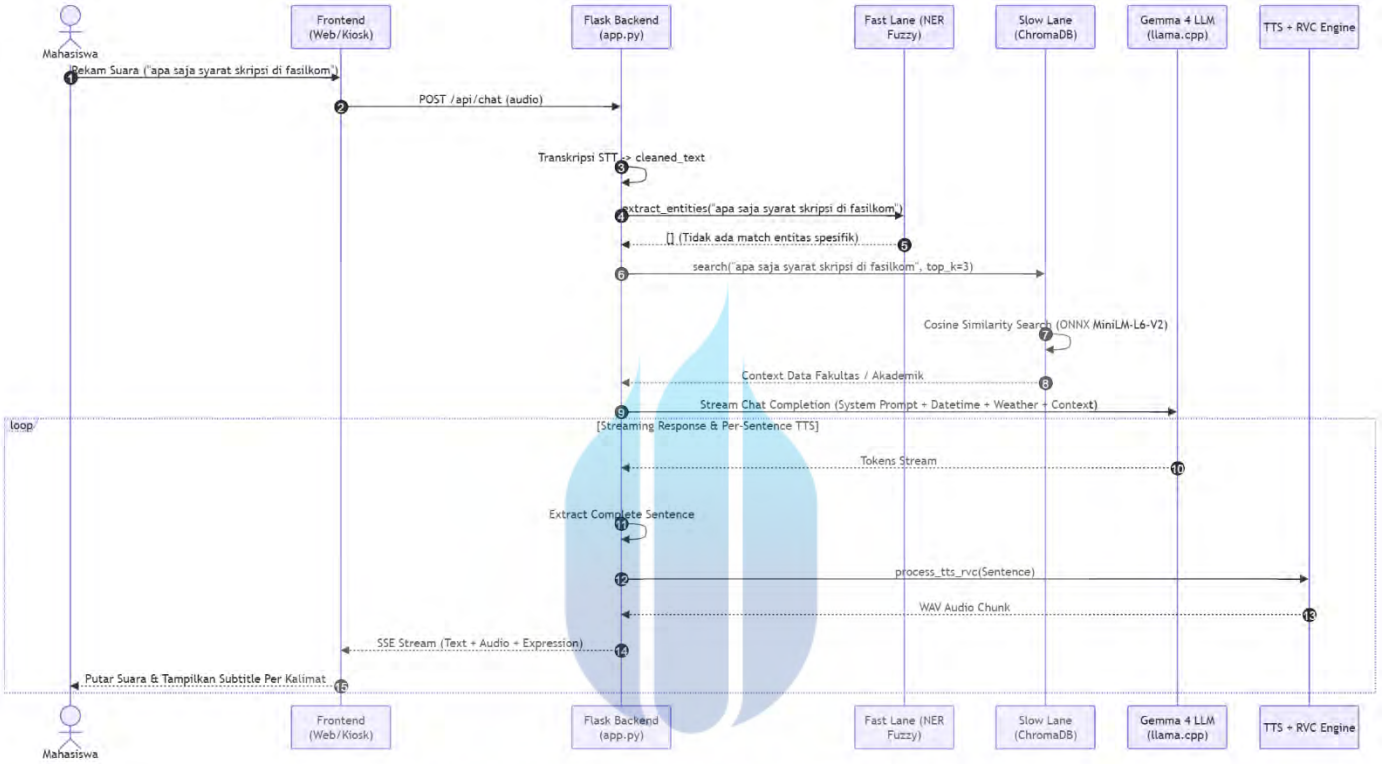
karakter) memerlukan kecocokan eksak (100), teks pendek (4-5 karakter) memerlukan skor 95, teks sedang (6-8 karakter) memerlukan skor 88, dan teks panjang memerlukan skor minimal 80. Alur pemrosesan entitas pada jalur cepat (*fast lane*) terjadi ketika kueri pengguna berhasil dipetakan secara langsung ke dalam basis data alias menggunakan teknik *fuzzy matching*. Urutan interaksi antarkomponen pada jalur cepat dari pengolahan input hingga penyerahan respons berbasis *template* ditunjukkan pada gambar 3.5 *sequence diagram fast lane.*



Gambar 3.5 *Sequence diagram fast lane*

Apabila satu alias merujuk pada lebih dari satu entitas unik, sistem menandai hasil pencocokan sebagai ambigu dan memicu mekanisme disambiguasi yang meminta klarifikasi kepada pengguna. Apabila jalur cepat tidak berhasil menemukan kecocokan, sistem beralih ke jalur lambat (*slow lane*) yang memanfaatkan ChromaDB sebagai *vector database* untuk

melakukan pencarian semantik berbasis *cosine similarity*. Hasil pencarian semantik digunakan sebagai konteks tambahan yang disuntikkan ke dalam prompt LLM, sehingga model bahasa dapat menghasilkan jawaban yang relevan meskipun entitas tidak berhasil dikenali melalui pencocokan string. Untuk alurnya dapat dilihat pada gambar 3.6 *sequence diagram slow lane*.



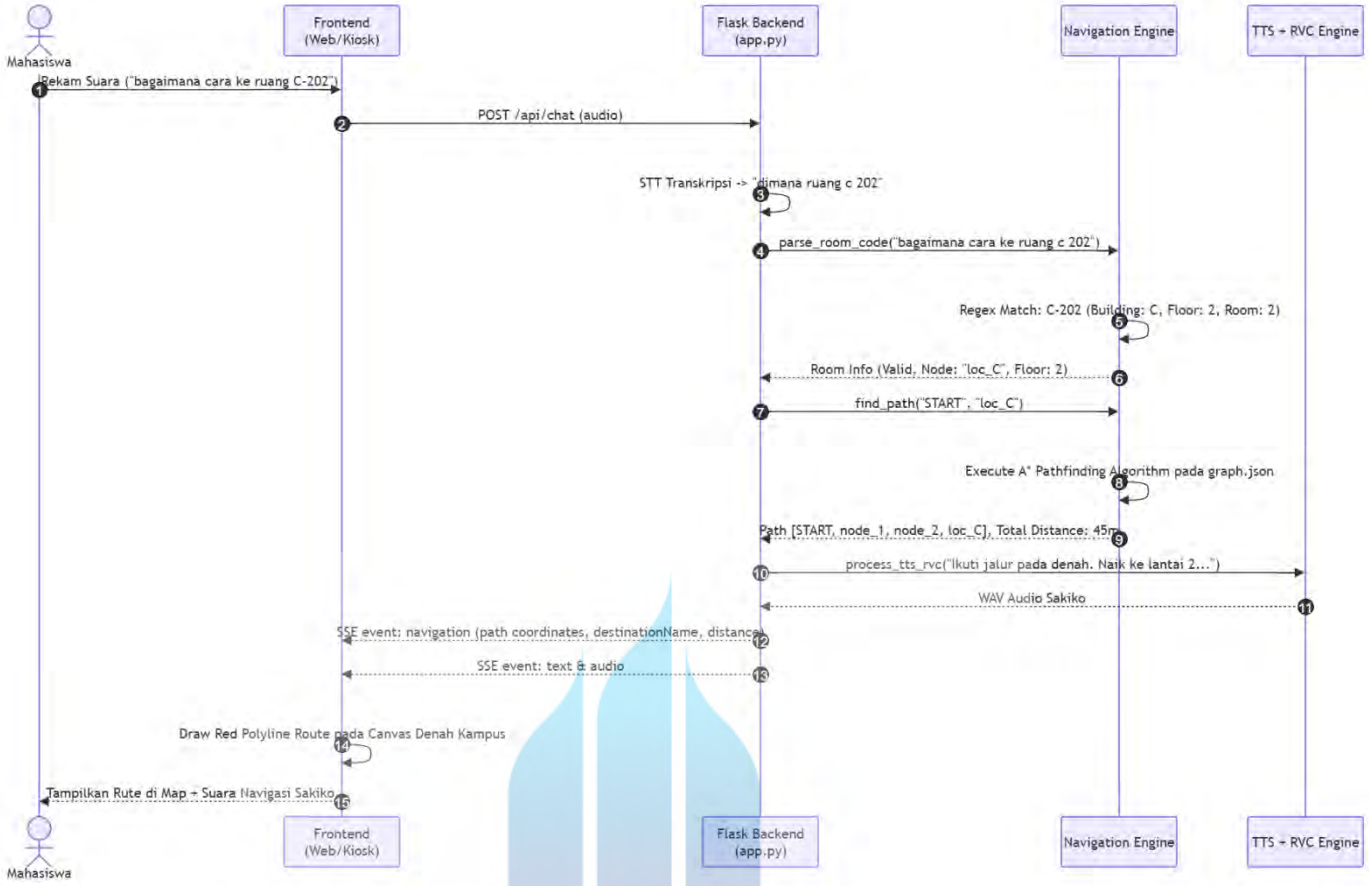
Gambar 3.6 *Sequence diagram slow lane*

Modul Hybrid NER pada penelitian ini menjalankan tugas pengenalan entitas bernama (*Named Entity Recognition*) menggunakan pendekatan berbasis pengetahuan (*knowledge-based*), bukan menggunakan model *sequence labeling* konvensional. Pendekatan ini dipilih karena seluruh entitas pada domain kampus, seperti nama dosen, tenaga kependidikan, fakultas, program studi, dan biro, telah terdefinisi secara lengkap dalam basis data JSON. Dengan demikian, pengenalan entitas dapat dilakukan melalui teknik pencocokan string tanpa memerlukan data pelatihan berlabel

3.3.4 Desain Navigasi Menggunakan Algoritma A*

Navigation Engine dirancang untuk menerjemahkan kueri navigasi pengguna menjadi rute visual pada denah kampus. Modul ini memiliki tiga mekanisme pengenalan tujuan yang dieksekusi secara berurutan. Pertama, deteksi kode ruangan menggunakan ekspresi reguler (*regex*) dengan pola \b(AD|BD|[A-I]|T)[\s\-]?(\d{3})\b yang mampu mengenali format kode ruangan seperti C-202, T-301, dan lainnya. Setiap kode ruangan yang terdeteksi divalidasi terhadap data gedung untuk memastikan kode gedung, nomor lantai, dan nomor ruangan valid. Kedua, pengecekan kueri ambigu untuk menangani kata kunci yang dapat merujuk pada lebih dari satu lokasi. Ketiga, pencocokan kata kunci (*keyword matching*) yang mencocokkan teks input dengan kamus pemetaan kata kunci ke lokasi.

Setelah tujuan berhasil diidentifikasi, sistem menjalankan algoritma A* untuk mencari rute terpendek dari titik awal asisten virtual kampus berada menuju lokasi tujuan. Graf navigasi terdiri dari simpul-simpul dengan koordinat spasial pada denah kampus dan sisi-sisi yang merepresentasikan jalur penghubung beserta jarak dalam meter. Fungsi heuristik yang digunakan adalah jarak Euclidean yang memenuhi syarat *admissible*. Hasil pencarian rute berupa daftar koordinat yang dikirim ke *frontend* untuk divisualisasikan pada gambar denah kampus, disertai teks respons yang menginformasikan jarak tempuh dan petunjuk lantai. Untuk lebih detailnya dapat dilihat pada diagram sequence pada gambar 3.7 *sequence diagram navigation*.



Gambar 3.7 *Sequence diagram navigation*

3.3.5 Metode Pengujian

Pengujian pada penelitian ini dilakukan menggunakan dua pendekatan untuk memastikan sistem asisten virtual kampus berjalan sesuai dengan rancangan yang telah dibuat. Dalam penelitian ini dilakukan 2 pengujian, yaitu :

1. Pengujian Fungsional (*Black Box Testing*)

   Pada pengujian pertama menggunakan metode pengujian *black box*, yaitu metode pengujian yang mengevaluasi *output* sistem asisten virtual kampus berdasarkan *input* yang diberikan. Tujuan dari pengujian metode ini ialah untuk memastikan agar sistem asisten virtual kampus dapat menjawab pertanyaan pengguna dengan benar dan bekerja dengan baik. Pada metode ini, sistem asisten virtual kampus akan diberi pertanyaan sebanyak 20 pertanyaan. Dengan skenario seperti tabel dibawah ini :

Tabel 3.1 Skenario pegujian *black box*

No.	Skenario	Tujuan Pengujian
1.	Pertanyaan faktual	Menguji ketepatan jawaban terhadap data profil dosen dan tenaga kependidikan, seperti jabatan, bidang ilmu, riwayat pendidikan, dan gelar akademik
2.	Jadwal mengajar	Menguji ketepatan sistem dalam menampilkan daftar jadwal mengajar dosen berdasarkan basis pengetahuan
3.	Lokasi dosen terkini	Menguji kemampuan sistem mencocokkan waktu saat ini dengan jadwal mengajar dosen untuk menentukan posisi ruangan secara real-time.
4.	Pertanyaan ambigu	Menguji kemampuan sistem bertanya balik (disambiguasi) ketika nama merujuk ke lebih dari satu orang, serta menguji pengenalan entitas dari sebagian nama saja.
5.	Pertanyaan kompleks	Menguji kemampuan sistem mengalihkan pertanyaan di luar template ke modul LLM untuk diproses.
6.	Tenaga kependidikan	Menguji pengenalan entitas staf tenaga kependidikan.
7.	Toleransi kesalahan transkripsi	Menguji ketangguhan *fuzzy string matching* dalam mengenali nama yang salah eja akibat kesalahan transkripsi suara (*Speech-to-Text*).
8.	Navigasi kampus	Menguji kemampuan sistem mengenali kueri navigasi dan menghasilkan rute menggunakan algoritma A* pada denah kampus.

Setiap pertanyaan saat pengujian dilakukan dokumentasi dalam bentuk tabel dengan kolom : *input* pengujian, *output* diharapkan, *output aktual*, dan hasilnya sesuai atau tidak sesuai.

2. Evaluasi Kinerja NER

   Pengujian kedua dilakukan untuk mengukur tingkat akurasi modul *Named Entity Recognition* (NER) secara statistik menggunakan dataset yang berisi 1.000 contoh pertanyaan terkait dosen, tenaga kependidikan, dan tanpa entitas. Untuk memastikan konsistensi hasil, pengujian dilakukan pada dua pengambilan sampel acak yaitu, 80/20 dan 70/30. Pengambilan sampel acak ini untuk membuktikan bahwa performa sistem tetap stabil meskipun diuji pada kumpulan pertanyaan berbeda-beda. Kinerja modul NER diukur menggunakan empat metrik evaluasi standar :

   a. *Accuracy* untuk mengukur persentase keseluruhan entitas yang berhasil dikenali dengan benar dari seluruh data yang diuji.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

   b. *Precision* untuk mengukur seberapa tepat sistem dalam mengenali entitas dari seluruh entitas yang berhasil dideteksi.

$$Precision = \frac{TP}{TP + FP}$$

   c. *Recall* untuk mengukur seberapa banyak entitas yang berhasil ditemukan oleh sistem dari seluruh entitas yang seharusnya dikenali.

$$Recall = \frac{TP}{TP + FN}$$

   d. *F1-Score* merupakan rata-rata harmonis antara *precision* dan *recall* yang memberikan gambaran performa sistem secara seimbang dalam satu nilai tunggal.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

   Keterangan :

   TP (*True Positive*) : Entitas yang benar dikenali oleh sistem.

   FP (*False Positive*) : Entitas yang salah dikenali oleh sistem.

FN (*False Negative*) : Entitas yang tidak berhasil dikenali oleh sistem.

3.3.6 Spesifikasi Perangkat

Berikut adalah spesifikasi perangkat keras dan perangkat lunak yang digunakan dalam penelitian ini.

Tabel 3.2 Spesifikasi computer

Komponen	Spesifikasi
Processor	Intel i5-11400F
GPU	NVIDIA GTX 1650 4GB
RAM	16 GB
Sistem Operasi	Windows

Tabel 3.3 Daftar perangkat lunak yang digunakan

Perangkat Lunak	Keterangan
Python 3.10	Bahasa pemograman utama
Flask	Framework backend web server
Gemma 4 E2B (Q4_K_M GGUF)	Model LLM untuk STT dan respons kompleks
llama.cpp	Framework inferensi LLM lokal
RapidFuzz	Library fuzzy string matching
ChromaDB	Vector database untuk semantic search
Edge-TTS	Text-to-Speech engine (id-ID-GadisNeural)
RVC v2	Voice conversion untuk suara karakter
Live2D Cubism SDK	Engine animasi karakter virtual
ffmpeg	Konversi dan pemrosesan audio

Dua tabel di atas menunjukkan spesifikasi komputer dan perangkat lunak yang digunakan dalam pengembangan sistem. Pemilihan perangkat lunak didasarkan pada kebutuhan masing-masing modul dalam sistem asisten virtual. Lalu untuk karakternya digambar terlebih dahulu menggunakan Adobe Photoshop, dapat dilihat pada gambar 3.8 ialah texture yang nanti akan disusun menjadi karakter utuh menggunakan Live2D Cubism Editor.

Gambar 3.8 Texture karakter