Nama : Diaz Mahardika
NIM 12030124140331
Mata Kuliah : Sistem Informasi Akuntansi – E
Dosen Pengampu : Dr. Totok Dewayanto, S.E., M.Si., Akt.

Desain dan Implementasi Prototipe AI Agent untuk Manajemen Rumah Sakit Menggunakan Google AI Studio dan Gemini AI "Medicore Ai"

1. Pendahuluan

- Manajemen Dokter dan Staf (*Doctor and Staff*): Mengelola jadwal dokter, informasi staf, dan alokasi sumber daya internal rumah sakit.
- Informasi Medis *(Medical Information*): Menyediakan informasi mengenai penyakit, gejala, perawatan, dan pengetahuan medis umum lainnya.
- Akuntansi Rumah Sakit *(Hospital Accounting*): Menangani fungsi keuangan seperti penagihan, pembayaran, penggajian, dan fungsi akuntansi lainnya.
- Manajemen Pasien *(Patient Management)*: Mengelola rekam medis pasien, data admisi, proses pemulangan, dan riwayat kesehatan.
- Penjadwalan Janji Temu (*Appointment Scheduling*): Menangani proses pemesanan, modifikasi, dan pembatalan janji temu dengan dokter.

Laporan ini akan menguraikan arsitektur sistem yang diusulkan, analisis mendalam terhadap salah satu modul fungsional, tumpukan teknologi yang dipilih, serta metodologi implementasi prototipe secara bertahap.

2. Arsitektur Konseptual AI Agent

Arsitektur modular yang terdefinisi dengan baik adalah syarat mutlak untuk skalabilitas dan pemeliharaan sistem. Kami mengadopsi praktik penggunaan *flowchart* dari analisis Sistem Informasi Akuntansi (SIA) untuk memastikan setiap aliran data dan interaksi sistem dipetakan secara eksplisit sebelum satu baris kode pun ditulis. Pendekatan ini memastikan desain yang terstruktur dan dapat dipertanggungjawabkan.

Pada tingkat tertinggi, arsitektur sistem ini berpusat pada modul inti bernama "Manage Hospital". Modul ini berfungsi sebagai Lapisan Orkestrasi dan Perutean (*Orchestration and Routing Layer*). Fungsi utamanya adalah untuk memparsing kueri dalam bahasa natural, mengklasifikasikan intensi pengguna, dan mendelegasikan tugas ke modulmodul fungsional hilir yang sesuai untuk dieksekusi. Kelima modul pendukung ini dirancang untuk menangani domain spesifik dalam operasional rumah sakit, memastikan setiap permintaan diproses oleh unit yang memiliki kapabilitas yang relevan.

Peran dari kelima modul khusus ini didefinisikan sebagai berikut:

Nama Modul	Fungsi Utama
Doctor And Staff	Mengelola data dan alokasi sumber daya manusia, termasuk jadwal dokter dan informasi staf di lingkungan rumah sakit.
Medical Information	Menyediakan akses ke basis pengetahuan medis umum, mencakup informasi tentang penyakit, gejala, dan opsi perawatan.
Hospital Accounting	Mengelola seluruh fungsi keuangan dan akuntansi, seperti penagihan pasien, penggajian staf, dan transaksi finansial lainnya.
Patient Management	Bertanggung jawab atas pengelolaan data pasien secara komprehensif, mulai dari rekam medis, admisi, hingga riwayat perawatan.
Appointment Scheduling	Menangani seluruh siklus janji temu pasien, termasuk proses pemesanan, perubahan jadwal, dan pembatalan.

Setiap modul dalam arsitektur ini dapat dianalisis lebih lanjut menggunakan kerangka kerja "blok pembangun sistem informasi" (*information system building block*) untuk memastikan setiap komponen dirancang secara komprehensif sebelum implementasi teknis.

3. Analisis Komponen Sistem: Studi Kasus Modul Akuntansi Rumah Sakit

Setiap modul fungsional dalam arsitektur AI Agent dapat diuraikan menjadi komponenkomponen dasar yang membentuk sistem informasi. Kerangka kerja "information system building block" yang digunakan dalam Sistem Informasi Akuntansi (SIA) menyediakan pendekatan terstruktur untuk menganalisis setiap elemen, mulai dari masukan data hingga mekanisme pengendalian. Dengan menggunakan modul Hospital Accounting sebagai studi kasus, kita dapat memetakan desain konseptualnya secara mendalam.

3.1 Masukan (Input Block)

Blok masukan bertanggung jawab untuk menangkap semua data yang relevan untuk diproses oleh sistem. Untuk modul Akuntansi Rumah Sakit, masukan utamanya adalah data transaksi keuangan yang beragam. Ini mencakup, namun tidak terbatas pada, data penagihan layanan medis kepada pasien, informasi penggajian dokter dan staf, transaksi pembelian aset atau bahan baku, serta detail pembayaran dari pasien atau pihak asuransi. Data ini dapat dimasukkan melalui berbagai media, mulai dari formulir digital hingga integrasi otomatis dengan sistem lain.

3.2 Model (Model Block)

Blok model adalah inti pemrosesan, di mana data masukan diolah menggunakan model *logicomathematical* untuk menghasilkan keluaran yang dikehendaki. Logika ini mengubah data mentah menjadi informasi yang bermakna. Sebagai contoh sederhana dalam konteks akuntansi, untuk menghasilkan laporan laba rugi, blok model akan menerapkan formula fundamental:

Laba = Pendapatan – Biaya

Model ini mengambil data pendapatan dari penagihan pasien dan data biaya dari penggajian, operasional, dan pembelian, kemudian mengolahnya untuk menghitung profitabilitas rumah sakit.

3.3 Keluaran (Output Block)

Blok keluaran adalah produk akhir dari sistem, yang menyajikan informasi bermutu kepada para pemangku kepentingan. Dalam modul Akuntansi Rumah Sakit, keluarannya dapat berupa berbagai dokumen dan laporan. Contohnya termasuk faktur (tagihan) yang dikirimkan kepada pasien, laporan keuangan periodik (seperti neraca dan laporan laba rugi) untuk manajemen, serta ringkasan penagihan yang dibutuhkan untuk klaim asuransi.

3.4 Teknologi (Technology Block)

Teknologi adalah alat penunjang yang memungkinkan seluruh komponen sistem berfungsi. Dalam fase prototyping proyek ini, tumpukan teknologi utama yang digunakan adalah Google AI Studio sebagai lingkungan pengembangan terpadu (IDE) berbasis web, dan Gemini API sebagai model pemrosesan inti yang menjalankan logika interpretasi dan pemrosesan permintaan pengguna.

3.5 Basis Data (Database Block)

Blok basis data berfungsi sebagai tempat penyimpanan data yang terorganisir. Dalam sistem informasi akuntansi berbasis komputer, interaksi data melibatkan beberapa jenis *file* yang berbeda. Untuk modul ini, kita dapat mengantisipasi penggunaan:

- File Induk (File Induk): Berisi data akun yang relatif statis, seperti daftar akun buku besar (misalnya, kas, piutang, utang) dan buku besar pembantu (misalnya, detail utang per pemasok).
- File Transaksi (File Transaksi): File sementara yang menyimpan catatan transaksi harian, seperti data penagihan baru atau pembayaran gaji, yang akan digunakan untuk memperbarui file induk.
- File Referensi (File Referensi): Menyimpan data standar yang digunakan untuk memvalidasi atau memproses transaksi, contohnya seperti daftar tarif layanan medis atau tabel kredit limit untuk pasien.
- File Arsip (File Arsip): Berisi catatan transaksi masa lalu yang dipertahankan untuk referensi di masa mendatang. Dalam konteks akuntansi rumah sakit, ini mencakup siklus penagihan historis, catatan tahun fiskal yang telah ditutup, dan informasi penggajian lampau untuk tujuan audit dan kepatuhan peraturan.

3.6 Pengendalian (Control Block)

Blok pengendalian sangat krusial untuk memastikan integritas, keamanan, dan keandalan sistem. Mengingat sensitivitas data keuangan, seluruh sistem informasi harus dilindungi dari berbagai ancaman, termasuk kebakaran, kecurangan, penggelapan, ketidakefisienan, dan sabotase. Mekanisme pengendalian dalam modul ini akan mencakup implementasi otorisasi akses berbasis peran dan, yang terpenting, memastikan pemeliharaan Jejak Audit Magnetis (*Magnetic Audit Trail*). Jejak audit ini mencatat setiap transaksi yang diproses oleh agen AI, menyediakan log yang tidak dapat diubah untuk tujuan akuntabilitas, analisis forensik, dan audit peraturan.Analisis ini menunjukkan bagaimana sebuah modul dapat dirancang secara sistematis. Selanjutnya, laporan ini akan membahas platform teknologi yang dipilih untuk merealisasikan prototipe ini.

4. Platform Prototyping dan Tumpukan Teknologi

Pemilihan Google AI Studio dan model Gemini sebagai tumpukan teknologi utama untuk fase prototyping didasarkan pada kombinasi strategis antara kecepatan pengembangan, aksesibilitas, dan kapabilitas canggih. Google AI Studio, sebagai *integrated development environment* (IDE) berbasis web yang gratis, berfungsi sebagai jalur cepat untuk mengubah ide menjadi aplikasi fungsional. Platform ini memberikan akses "raw" atau mentah ke modelmodel Gemini, yang memungkinkan pengembang untuk berinteraksi dengan kemampuan inti model secara presisi. Akses tanpa filter ini sangat penting untuk prototyping tingkat perusahaan, karena memungkinkan kami untuk melewati pagar pembatas percakapan produk yang ditujukan untuk konsumen dan secara langsung memodelkan perilaku deterministik dan dapat diprediksi yang diperlukan untuk aplikasiaplikasi krusial.

Peran strategis Google AI Studio dalam proyek ini terletak pada kemampuannya untuk mempercepat siklus eksperimen dan iterasi. Pengembang dapat dengan cepat merancang, menguji, dan menyempurnakan perilaku agen AI sebelum menulis satu baris pun kode produksi.

Kapabilitas utama Google AI Studio yang relevan untuk pengembangan prototipe ini adalah sebagai berikut:

- Antarmuka Prompt Terpadu: Platform ini menyediakan antarmuka *chat* yang memungkinkan pembuatan *prompt* dengan System Instructions. Fitur ini sangat penting untuk mendefinisikan persona, batasan, dan perilaku dasar agen AI, memastikan respons yang konsisten dan sesuai dengan tujuan aplikasi.
- Pengaturan Eksekusi (Run Settings): Pengembang diberikan kontrol granular terhadap parameter eksekusi model. Salah satu parameter terpenting adalah Temperature, yang dapat disesuaikan untuk mengatur tingkat kreativitas atau determinisme respons. Nilai Temperature yang rendah cocok untuk tugas yang membutuhkan akurasi faktual, sementara nilai yang tinggi lebih sesuai untuk tugas kreatif.
- Kapabilitas Lanjutan: AI Studio mengintegrasikan fiturfitur canggih yang krusial untuk membangun agen yang fungsional. Fitur Function Calling memungkinkan model untuk terhubung dan berinteraksi dengan API eksternal, sedangkan Code Execution memberikan kemampuan kepada model untuk menjalankan kode (misalnya Python) untuk menyelesaikan tugastugas komputasi atau analisis data yang kompleks.
- Fitur "Get Code": Setelah prototipe divalidasi di dalam antarmuka web, fitur ini menjadi akselerator pengembangan yang sangat kuat. Dengan satu klik, AI Studio dapat menghasilkan cuplikan kode siap pakai dalam berbagai bahasa pemrograman populer (seperti Python, Node.js, dan Swift) yang mereplikasi seluruh konfigurasi prototipe, termasuk instruksi sistem, parameter model, dan definisi panggilan fungsi.

Dengan kapabilitas ini, Google AI Studio menyediakan lingkungan yang ideal untuk merancang dan memvalidasi agen AI manajemen rumah sakit, sebelum beralih ke fase pengembangan aplikasi yang sesungguhnya.

5. Metodologi Implementasi Prototipe

Pembangunan prototipe AI Agent di Google AI Studio dilakukan melalui pendekatan bertahap yang terstruktur. Metodologi ini memungkinkan pengembangan iteraktif, dimulai dari definisi perilaku inti agen hingga integrasinya dengan sistem eksternal, memastikan setiap fungsi divalidasi sebelum melangkah ke tahap berikutnya.

5.1 Tahap 1: Definisi Persona dan Instruksi Sistem

Langkah pertama adalah membentuk persona inti dari AI Agent menggunakan fitur System Instructions di Google AI Studio. Fitur ini berfungsi sebagai arahan fundamental yang akan memandu seluruh respons model. Serupa dengan analogi pembuatan chatbot alien yang diinstruksikan untuk "berkomunikasi seolaholah berasal dari Europa, salah satu bulan Jupiter", persona untuk agen manajemen rumah sakit ini dirancang untuk menjadi "profesional, efisien, dan komunikatif". Instruksi sistem yang terperinci memastikan bahwa agen secara konsisten mempertahankan nada yang sesuai dan fokus pada tugastugas yang relevan.5.2 Tahap 2: Prototyping Fungsi Inti Melalui Chat

Setelah persona ditetapkan, fungsi inti dari setiap modul diuji secara interaktif melalui antarmuka *chat*. Proses ini melibatkan pemberian serangkaian *prompt* yang merepresentasikan permintaan pengguna di dunia nyata. Sebagai contoh, untuk menguji modul "Appointment Scheduling", seorang pengembang dapat memberikan *prompt* seperti:

"Saya ingin membuat janji temu dengan Dr. Budi pada hari Selasa pagi."

Respons yang diharapkan dari model adalah pemahaman terhadap entitas kunci (nama dokter, hari, waktu) dan permintaan klarifikasi atau konfirmasi langkah selanjutnya. Tahap ini krusial untuk menyempurnakan kemampuan model dalam memahami bahasa natural dan mengekstrak informasi yang relevan dari permintaan pengguna.

5.3 Tahap 3: Integrasi dengan Sistem Eksternal Menggunakan *Function Calling*

Function Calling adalah mekanisme yang mengubah agen dari sistem yang murni informasional (terbatas pada data pelatihannya) menjadi sistem aksi yang transaksional dan stateful. Fitur ini memungkinkan model Gemini untuk mengeksekusi operasi di dunia nyata dan berinteraksi dengan sumber data langsung, sehingga mendobrak batasan basis pengetahuan statisnya. Ketika menerima permintaan yang memerlukan data eksternal, model dapat memicu fungsi yang telah didefinisikan, seperti "memeriksa ketersediaan jadwal dr. Budi di database" atau "mengambil riwayat medis pasien dengan ID tertentu".

5.4 Tahap 4: Transisi dari Prototipe ke Kode Aplikasi

Setelah seluruh perilaku agen, termasuk panggilan fungsi, telah divalidasi dalam lingkungan AI Studio, langkah terakhir adalah mentransisikan prototipe tersebut menjadi kode aplikasi yang sesungguhnya. Fitur "Get code" secara signifikan menyederhanakan proses ini. Fitur ini secara otomatis menghasilkan cuplikan kode (misalnya, dalam Python) yang mereplikasi seluruh konfigurasi prototipemulai dari *System Instructions*, parameter model seperti Temperature, hingga skema *Function Calling*. Dengan menyediakan kode dasar yang sudah terstruktur, fitur ini secara drastis mengurangi waktu dan upaya yang diperlukan pengembang untuk mengintegrasikan logika AI ke dalam aplikasi backend mereka.

Kode yang dihasilkan dari tahap ini menjadi fondasi untuk pengembangan lebih lanjut, yang memerlukan manajemen versi dan kolaborasi tim yang solid.

6. Manajemen Kode dan Kolaborasi Tim

Dalam pengembangan proyek perangkat lunak, terutama yang melibatkan kecerdasan buatan, penggunaan sistem kontrol versi yang andal dan platform kolaborasi menjadi sangat penting. Saat tim pengembang bekerja sama dalam sebuah proyek, alat seperti Git dan GitHub memastikan bahwa perubahan kode dapat dilacak, dikelola, dan diintegrasikan secara sistematis tanpa menimbulkan konflik.

Git adalah sebuah *version control system* yang berfungsi di komputer lokal, bertugas melacak setiap perubahan yang terjadi pada file kode. Dengan Git, pengembang dapat menyimpan "snapshot" dari proyek pada titik waktu tertentu dan melihat riwayat versi sebelumnya. Di sisi lain, GitHub adalah platform berbasis *cloud* yang dibangun di atas Git. GitHub menyediakan layanan untuk menyimpan *repository* (wadah proyek) secara online, sehingga memungkinkan pengembang untuk menyimpan kode mereka secara terpusat, berbagi dengan anggota tim lain, dan mengelola proyek secara kolaboratif.

Alur kerja tim untuk proyek ini akan memanfaatkan kedua alat tersebut secara sinergis. Kode yang dihasilkan dari fitur "Get Code" di Google AI Studio akan menjadi basis awal yang kemudian disimpan dan dikelola dalam sebuah *repository* di GitHub. Konsepkonsep utama GitHub berikut akan menjadi fundamental dalam alur kerja kolaboratif tim:

1. Repository (Repo): Setiap proyek AI Agent akan ditempatkan dalam *repository*nya sendiri di GitHub. *Repo* ini akan berfungsi sebagai wadah pusat untuk semua file kode, dokumentasi teknis, dan seluruh riwayat perubahan, menjadi satusatunya sumber kebenaran (*single source of truth*) bagi seluruh tim.
2. Branch: Untuk menjaga stabilitas kode utama, setiap pengembang akan membuat *branch* (cabang) terpisah untuk mengerjakan fitur atau modul baru. Sebagai contoh, pengembang yang mengimplementasikan modul Medical Information akan membuat *branch* baru bernama feature/medicalinfo. Strategi percabangan ini mengisolasi aliran pengembangan, mencegah regresi, dan memastikan *branch* main tetap menjadi sumber tunggal untuk kode yang stabil dan siap dideploy setiap saat.
3. Push dan Pull: Kolaborasi tim dijaga melalui perintah push dan pull. Setelah seorang pengembang menyelesaikan pekerjaannya di *branch* lokal, ia akan mengirimkan (push) perubahannya ke *repository* di GitHub. Sebaliknya, untuk memastikan mereka bekerja dengan versi kode terbaru dari anggota tim lain, mereka akan secara rutin mengambil (pull) pembaruan dari *repository* pusat. Mekanisme ini menjaga agar semua anggota tim tetap sinkron.

Dengan menerapkan alur kerja ini, proses pengembangan menjadi lebih terstruktur, transparan, dan efisien, yang merupakan kunci keberhasilan dalam proyek pengembangan perangkat lunak kolaboratif.

Fungsifungsi utama yang akan dikelola oleh prototipe AI Agent ini mencakup:

7. Kesimpulan

Laporan teknis ini telah menunjukkan bahwa kombinasi antara metodologi desain sistem yang terstruktur yang diadopsi dari prinsipprinsip Sistem Informasi Akuntansi dengan platform prototyping AI modern seperti Google AI Studio, menyediakan jalur yang efektif dan efisien untuk merancang agen manajemen rumah sakit yang canggih. Pendekatan hibrida ini secara signifikan mengurangi risiko implementasi AI dengan mendasarkannya pada pemikiran sistem yang sudah mapan, menyediakan metodologi yang jelas dan dapat dipertahankan untuk membangun agen yang kompleks dan krusial bagi misi organisasi.

Link Github Apps Preview :

https://github.com/diazmahardika25/Rumah-Sakit-Indah-Kac.git