APLIKASI RESERVASI PASIEN DAN REKAM MEDIS BIDAN MANDIRI DENGAN GOLANG SEBAGAI BACKEND

Zaidan Luthfi
Program Studi S1 Teknik Komputer
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
zaidanluthfi@student.telkomuniversity.ac.id

Faisal Candrasyah Hasibuan
Program Studi S1 Teknik Komputer
Fakultas Teknik Elektro
Universitas Telkom
Bandung, Indonesia
faicanhasfcb@telkomuniversity.ac.id

***Abstrak* — Bidan memainkan peran penting dalam layanan kesehatan ibu, anak, dan keluarga berencana. Namun, pencatatan layanan masih menggunakan metode konvensional yang tidak efisien, meningkatkan beban kerja, dan risiko kehilangan data. Untuk mengatasi masalah ini, dikembangkan sistem informasi Ninasys, yang terdiri dari dua aplikasi website: satu untuk bidan dan satu untuk pasien. Aplikasi ini menggunakan React.js dan Next.js untuk *frontend*, Golang untuk *backend*, dan MongoDB sebagai *database*. Ninasys memfasilitasi pencatatan medis bidan dan reservasi pasien, dengan pengembangan berbasis model *waterfall*. Pengujian menunjukkan bahwa sistem ini mampu mengurangi waktu reservasi dan pencatatan medis hingga 30% dan mendapatkan tingkat kepuasan pengguna sebesar 84.2%.**

Kata kunci Rekam medis elektronik, pelayanan kesehatan, pengembangan web, sistem informasi

I. PENDAHULUAN

Dalam era digital saat ini, kemajuan teknologi informasi telah berdampak pada berbagai sektor, termasuk pelayanan kesehatan. Salah satu aspek penting dalam layanan kesehatan adalah pengelolaan rekam medis, yang berfungsi mencatat dan menyimpan informasi medis pasien seperti data pribadi, hasil pemeriksaan, dan tindakan pengobatan. Data ini sangat penting untuk memberikan pelayanan kesehatan yang berkesinambungan dan berkualitas[1]. Meski demikian, banyak praktik bidan masih mengandalkan metode konvensional untuk mencatat dan mengelola rekam medis. Pendekatan manual ini sering kali mengakibatkan data yang tidak tersusun dengan baik dan meningkatkan risiko kesalahan pencatatan.

Dalam praktik bidan, akurasi dan keteraturan data sangatlah penting. Namun, penggunaan metode manual meningkatkan risiko kehilangan data, kesalahan penulisan, dan memperlambat proses pencarian data. Sebuah studi di Rumah Sakit Universitas Airlangga menemukan bahwa terdapat masalah keterlambatan dalam pengembalian berkas rekam medis di unit rawat jalan dan rawat inap, dengan persentase keterlambatan mencapai 48% pada tahun 2016 dan 56% pada tahun 2017 [2]. Situasi ini memperumit pengelolaan data pasien, yang seharusnya dapat diakses dengan cepat dan akurat.

Kesalahan dalam rekam medis bidan dapat berdampak serius pada kesehatan pasien. Sebagai contoh, jika rekam medis tidak mencatat riwayat alergi, penyakit kronis, atau obat yang sedang dikonsumsi pasien, bidan dapat memberikan perawatan yang tidak sesuai dengan kondisi pasien. Sebuah studi yang dipublikasikan di JAMA Network Open pada tahun 2020 menemukan bahwa 21,4% pasien melaporkan setidaknya satu kesalahan dalam catatan perawatan mereka, dan 40,5% dari kesalahan tersebut dianggap serius. Kesalahan yang paling umum meliputi kesalahan pengobatan, informasi yang salah atau tidak lengkap, serta data yang tidak diperbarui [3]. Hal ini dapat berakibat pada reaksi alergi, interaksi obat yang berbahaya, atau bahkan kematian.

II. KAJIAN TEORI

A. Bidan Mandiri

Bidan mandiri adalah tenaga kesehatan yang memiliki wewenang untuk memberikan layanan kebidanan secara independen kepada masyarakat. Mereka beroperasi secara mandiri tanpa keterikatan dengan institusi kesehatan tertentu dan bertanggung jawab penuh atas praktik yang dilakukan. Dalam perannya, bidan mandiri menyediakan layanan kesehatan ibu dan anak, termasuk perawatan selama kehamilan, persalinan, masa nifas, serta layanan kesehatan reproduksi. Umumnya, bidan mandiri membuka klinik sendiri atau menjalankan praktik di rumah untuk melayani pasien.

Sistem informasi sangat membantu bidan mandiri dalam mengelola operasional harian secara lebih efisien. Dengan sistem ini, berbagai proses seperti pencatatan rekam medis, pengelolaan jadwal konsultasi, dan administrasi pasien dapat dilakukan secara digital. Hal ini mengurangi potensi kesalahan manual, mempercepat pemrosesan data, dan memungkinkan bidan untuk lebih fokus pada pelayanan kesehatan dibandingkan administrasi.

B. *Golang*

Golang (Go *Language*) merupakan sebuah bahasa pemrograman yang mengkombinasikan keamanan dan performa [4] untuk pengembangan sistem yang bersifat open source dan dikembangkan di Google oleh Rob Pike, Robert Griesemer, dan Ken Thompson [5] beserta kontributor lainnya dalam komunitas pengembang *open source* . Hingga saat ini, Golang mulai banyak digunakan pada perusahaan-perusahaan besar maupun *startup* yang bergerak di bidang teknologi. Hal ini dikarenakan pemrograman dengan Golang ini memiliki beberapa kelebihan, antara lain [4][5][6]: (1) Mendukung konkurensi dalam sistem pemrograman dengan sangat baik dengan pengaplikasiannya sendiri yang cukup mudah. (2) Merupakan bahasa pemrograman yang bersifat open source. (3) Memiliki sistem *garbage collection* yang baik dengan memanfaatkan bantuan *built-in garbage collector process* (Goroutines). (4) Memiliki sintaks yang bersifat bersih, tidak mengotori sistem terlalu berlebihan. (5) Bahasa pemrograman yang *reliable* dan cepat dalam skala besar.

C. *MongoDB*

MongoDB merupakan sebuah *database* NoSQL yang dirancang untuk menyimpan, dan mengelola data dalam format *document-oriented* atau berbasis dokumen [7]. Berbeda dengan sistem manajemen basis data relasional konvensional yang menggunakan tabel, MongoDB menggunakan struktur data fleksibel berupa dokumen yang disebut BSON (*Binary* JSON), yang memungkinkan penyimpanan data dinamis dan tidak terstruktur secara kaku [8]. MongoDB sangat cocok diterapkan dalam pengembangan aplikasi modern yang memerlukan skalabilitas tinggi dan kemampuan untuk menangani data dalam volume besar dengan beragam format [9].

Dalam penerapannya, MongoDB sering digunakan pada sistem yang memerlukan pengolahan data yang cepat dan efisien, seperti *big data analytics*, *content management systems* (CMS), dan aplikasi yang membutuhkan penyimpanan data real-time [7]. MongoDB juga mendukung fitur replikasi dan sharding, yang menjamin ketersediaan data dan memungkinkan pengelolaan skala horizontal sesuai kebutuhan. Sistem ini sangat berguna dalam mengurangi kompleksitas pengelolaan data, mempercepat proses pengambilan keputusan bisnis, serta memungkinkan pengembang untuk lebih fokus pada pengembangan fitur aplikasi tanpa perlu khawatir terkait masalah skalabilitas atau performa *database* [10].

III. METODE

A. *Analisis Kebutuhan*

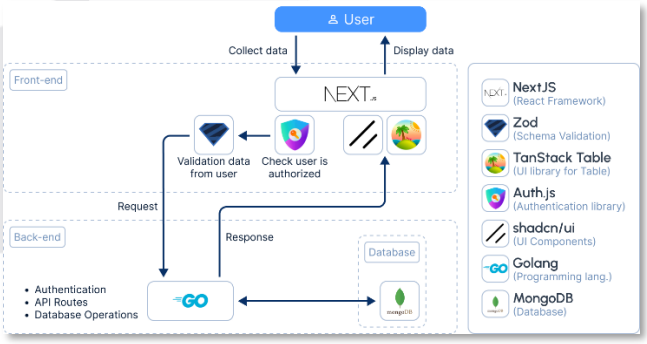
Sebelum pengembangan website rekam medis dimulai, telah dilakukan tahap wawancara dengan calon pengguna aplikasi untuk mengidentifikasi spesifikasi kebutuhan yang diperlukan. Dari wawancara ini, ditemukan beberapa kebutuhan utama yang harus dipenuhi oleh aplikasi. Pertama, pengguna harus dapat melakukan reservasi secara online, yang memungkinkan mereka untuk memesan layanan kesehatan dengan mudah. Kedua, aplikasi harus menyediakan fitur pengingat bagi pasien dan bidan terkait layanan reservasi, untuk memastikan kelancaran dan ketepatan waktu layanan tersebut.

Selain itu, aplikasi ini perlu mendukung sistem login yang memungkinkan akses dengan beberapa peran berbeda. Terdapat tiga peran utama yang akan diimplementasikan dalam aplikasi ini, yaitu pasien, admin, dan superadmin. Setiap peran ini memiliki fungsionalitas yang spesifik. Misalnya, pengguna dengan peran bidan diberikan wewenang untuk melakukan berbagai tindakan terkait layanan medis, termasuk membuat, menampilkan, mengubah, dan menghapus data pasien. Bidan juga harus dapat melakukan pencarian berdasarkan data pasien untuk mempermudah proses pelayanan.

Sementara itu, pengguna dengan peran superadmin memiliki akses lebih luas, termasuk kemampuan untuk melakukan rekapitulasi data dan mengekspor informasi tersebut ke dalam format Excel. Aplikasi ini juga harus memiliki kemampuan untuk menampilkan jumlah pasien dalam jangka waktu tertentu untuk analisis dan perencanaan layanan medis. Dengan memenuhi semua kebutuhan ini, aplikasi rekam medis diharapkan dapat meningkatkan efisiensi dan kualitas pelayanan kesehatan secara keseluruhan.

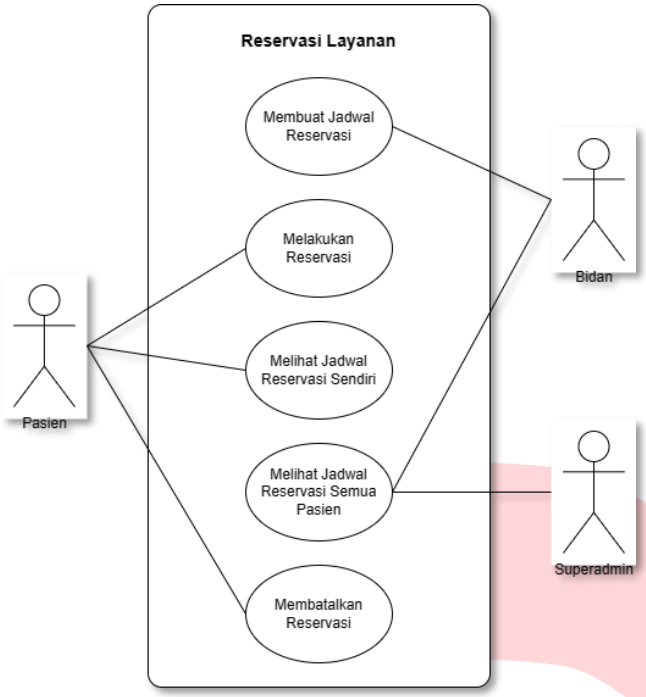
B. *Desain Sistem dan Arsitektur*

Sistem *backend* aplikasi ini dibangun menggunakan bahasa pemrograman Golang, yang dikenal karena kecepatan dan efisiensinya dalam menangani proses skala besar. Pada sisi *backend*, Golang berperan penting dalam menjalankan berbagai proses seperti autentikasi, pengelolaan rute API, dan operasi *database*. Ketika permintaan diterima dari *frontend*, pertama-tama akan dilakukan autentikasi untuk memastikan pengguna memiliki akses yang sah menggunakan Auth.js. Setelah autentikasi, data akan divalidasi dengan untuk menjaga integritas dan keamanan sistem. *Backend* kemudian akan menangani logika bisnis dan mengatur interaksi dengan *database*. Untuk penyimpanan data, sistem ini menggunakan MongoDB, sebuah NoSQL *database* yang fleksibel dan skalabel, cocok untuk menangani data dalam jumlah besar serta beragam jenis dokumen. Setelah operasi basis *database*. Setelah data diolah, hasilnya akan dikembalikan ke *frontend* untuk ditampilkan kepada pengguna.



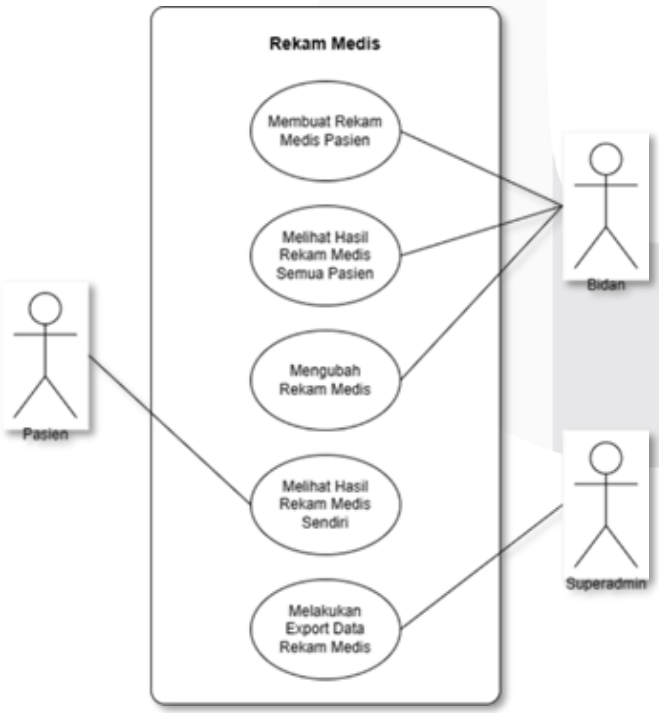
GAMBAR 1 Sistem Desain Aplikasi

Selain itu, rancangan Use Case Diagram juga dibuat. Penggunaan Use Case Diagram sering digunakan untuk memberikan gambaran sebagian atau keseluruhan sistem dalam bentuk yang esensial dan mengomunikasikan ruang lingkup dari suatu proyek yang sedang di buat [31]. Berikut merupakan berbagai Use Case Diagram yang digunakan untuk aplikasi Rekam Medis Bidan Mandiri.

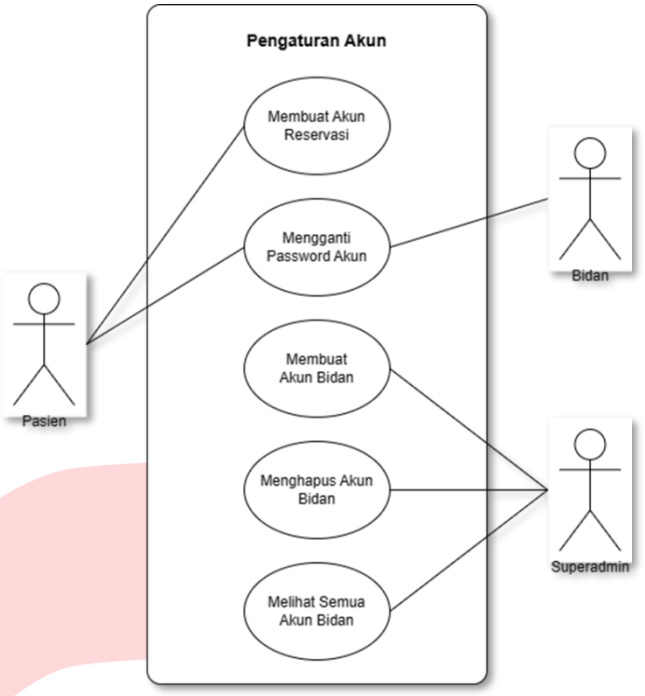


GAMBAR 2 Use Case Diagram Bagian Reservasi Layanan

Pada Use Case Diagram bagian Reservasi Layanan di atas Pasien dapat melakukan reservasi, melihat jadwal reservasi sendiri dan membatalkan kegiatan reservasi. Bidan pada bagian ini dapat membuat jadwal reservasi, dan melihat jadwal reservasi untuk semua pasien. Sedangkan role Superadmin pada Use Case Diagram ini hanya bisa melihat jadwal reservasi untuk semua pasien.



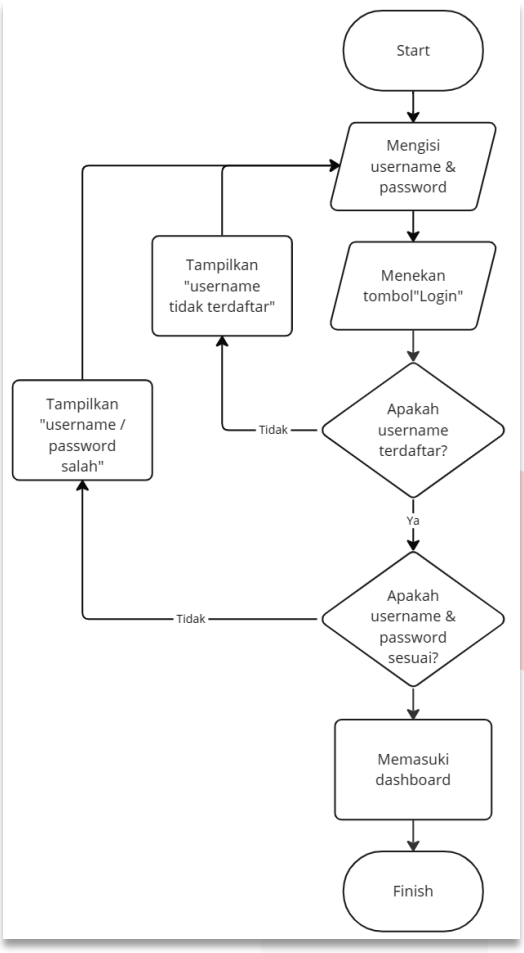
GAMBAR 3 Use Case Diagram Bagian Rekam Medis



GAMBAR 4 Use Case Diagram bagian Pengaturan Akun

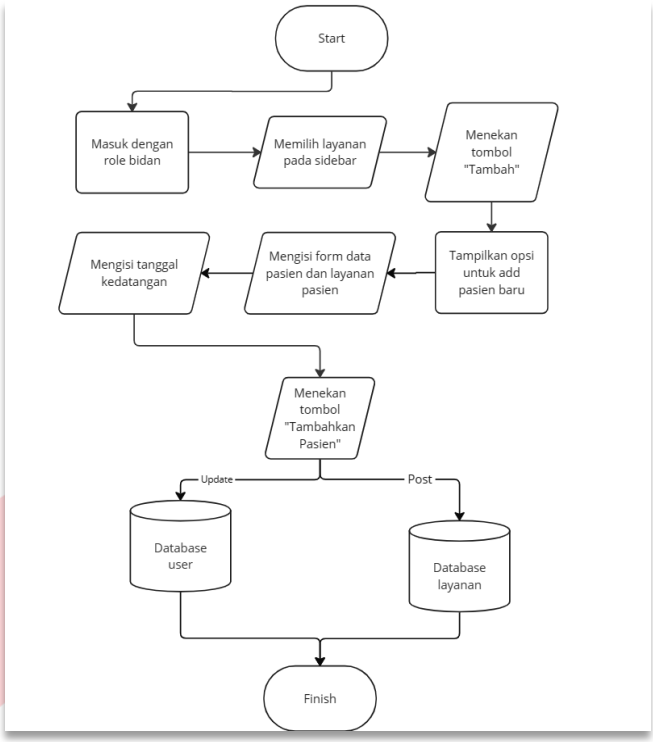
Pada Use Case Diagram bagian Pengaturan Akun Pasien dapat membuat akun untuk melakukan reservasi dan mengubah password dari akun mereka. Bidan hanya dapat mengganti password akun. Sedangkan Superadmin dapat membuat akun untuk Bidan, menghapus akun Bidan, dan melihat semua akun Bidan.

Selain itu, terdapat flowchart yang menggambarkan alur masuk pengguna pada semua role, terutama untuk bidan dan superadmin. Proses dimulai dengan pengguna memasukkan "username" dan "password". Jika informasi login yang dimasukkan tidak sesuai atau pengguna tidak terdaftar, sistem akan menampilkan pesan kesalahan pada halaman login. Namun, jika data login valid, pengguna akan langsung diarahkan ke halaman dashboard sesuai dengan role mereka.



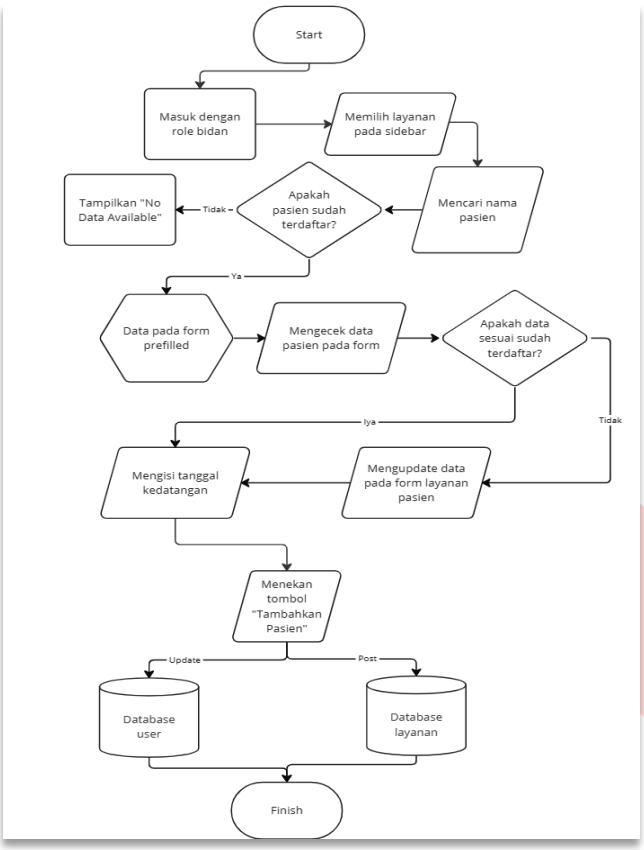
GAMBAR 5 Flowchart Login untuk Semua Role

Saat bidan menangani pasien baru, proses dimulai dengan memilih layanan yang telah ditentukan oleh pasien melalui reservasi sebelumnya. Setelah itu, bidan akan membuat data pasien baru dengan mengisi informasi yang diperlukan secara lengkap. Data pasien yang telah diisi kemudian akan disimpan dalam dua *database* utama, yaitu "Database user" untuk menyimpan informasi pasien, dan "Database layanan" untuk mencatat rincian layanan yang diberikan.



GAMBAR 6 Flowchart Layanan Pasien Baru

Untuk menangani layanan bagi pasien lama, bidan akan memilih layanan yang sesuai dengan reservasi yang telah dilakukan oleh pasien. Setelah itu, bidan akan mencari apakah pasien sudah terdaftar dalam sistem atau belum. Jika pasien belum terdaftar, sistem akan menampilkan pemberitahuan di halaman layanan. Namun, jika pasien sudah terdaftar, bidan dapat memperbarui data pasien yang ada. Setelah diperbarui, data pasien akan disimpan kembali ke dalam "Database user" dan "Database layanan".



GAMBAR 7 Flowchart pada Layanan Pasien Lama

IV. HASIL DAN PEMBAHASAN

Dalam proyek ini, telah dikembangkan *backend* dengan arsitektur yang dibangun menggunakan bahasa pemrograman Go (Golang). Golang. *backend* ini memiliki enam belas (16) API endpoint yang dirancang untuk mengelola berbagai fungsi dan layanan, yang memungkinkan interaksi pengguna yang beragam dan modular serta memudahkan pemeliharaan dan pengembangan lanjutan.

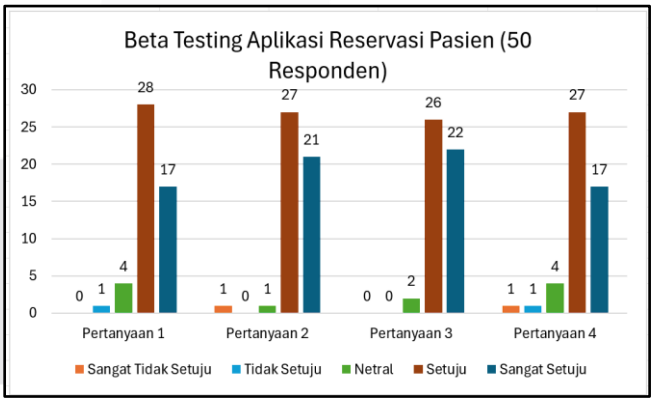
TABEL 1 List API Endpoint

No	Endpoint
1	/api/allsoap
2	/api/bidanlogin
3	/api/chart
4	/api/count
5	/api/countanually
6	/api/deletebidan
7	/api/edit
8	/api/export
9	/api/findpasien
10	/api/getbidan
11	/api/getreservasi
12	/api/input
13	/api/registbidan
14	/api/reservasi
15	/api/soap
16	/api/table

Database yang digunakan untuk backend ini adalah MongoDB, sebuah *database* NoSQL yang dikenal karena fleksibilitasnya dalam penanganan data tanpa skema yang ketat. Untuk implementasi dan *deployment*, *backend* Golang di-hosting menggunakan Google Cloud Run. Google Cloud Run menyediakan lingkungan *serverless* yang mendukung skalabilitas otomatis, sehingga *backend* dapat menyesuaikan kapasitasnya secara dinamis sesuai dengan beban kerja yang ada. Selain itu, MongoDB di-*hosting* menggunakan layanan MongoDB Atlas, sebuah platform *database-as-a-service* yang sepenuhnya dikelola oleh penyedia layanan. MongoDB Atlas menawarkan fitur-fitur seperti otomatisasi dalam deployment, skala yang dapat disesuaikan dengan kebutuhan, serta pemantauan kinerja secara *real-time*.

A. *Dampak Aplikasi*

Hasil pengujian aplikasi menunjukkan tingkat kepuasan pengguna yang cukup tinggi terkait tampilan antarmuka dan fitur yang tersedia. Dari umpan balik yang diterima, disimpulkan bahwa informasi yang disajikan oleh aplikasi mudah dipahami dan tidak membingungkan. Data yang dianalisis dari formulir tanggapan menunjukkan bahwa sebagian besar dari pengguna merasa puas dengan berbagai aspek aplikasi. Persentase tanggapan mengenai berbagai fitur dan tampilan aplikasi berada dalam kategori "Setuju" dan "Sangat Setuju" di berbagai aspek pengujian. Secara keseluruhan, rata-rata tingkat kepuasan mencapai sekitar 82%, yang merupakan indikasi kuat bahwa mayoritas pengguna merasa aplikasi ini telah memenuhi kebutuhan mereka dengan baik.



GAMBAR 8 Hasil Beta Testing Aplikasi Reservasi Pasien

B. *Uji Waktu Respons API*

Proses pengujian dilakukan menggunakan Apache JMeter untuk mengatur simulasi beban yang sesuai dengan tingkat beban yang telah ditentukan: 5 dan 30 user per menit. Tujuan utama dari pengujian ini adalah untuk memverifikasi bahwa API dapat menangani permintaan dengan efektif di bawah kondisi beban yang bervariasi. Jmeter juga akan digunakan untuk memantau dan mencatat waktu selama pengujian, yang akan membantu dalam analisis kinerja API.

Pengujian pada tingkat beban normal menunjukkan variasi waktu respons API yang signifikan di antara berbagai endpoint. Misalnya, endpoint "/api/bidanlogin" memiliki waktu respons rata-rata tertinggi dengan 1404 ms, menunjukkan adanya potensi bottleneck atau proses yang berat pada endpoint ini. Selain itu, endpoint seperti "/api/export" dan "/api/soap" juga menunjukkan waktu respons yang tinggi, dengan nilai maksimum masing-masing mencapai 1842 ms dan 3703 ms. Di sisi lain, beberapa endpoint seperti "/api/chart", "/api/count", dan "/api/deletebidan" memiliki waktu respons yang lebih rendah, dengan rata-rata di bawah 1000 ms. Secara keseluruhan, waktu respons rata-rata untuk semua endpoint adalah 1013 ms, yang menunjukkan bahwa sebagian besar endpoint memiliki waktu respons yang cukup baik pada tingkat beban normal.

TABEL 2 Tingkatan Beban Normal

Endpoint	Average (ms)	Min (ms)	Maximum (ms)
/api/allsoap	860	377	1958
/api/bidanlogin	1404	680	2002
/api/chart	619	324	1073
/api/count	984	340	1923
/api/countanually	754	323	1079
/api/deletebidan	780	291	1079
/api/edit	732	304	1019
/api/export	1014	352	1842
/api/findpasien	1000	361	1093
/api/getbidan	964	329	1907
/api/getreservasi	863	513	1063
/api/input	1048	346	1936
/api/registbidan	762	304	1061
/api/reservasi	1227	418	1976
/api/soap	1647	871	3703
/api/table	1132	384	1972
Rata-rata	1013	391	1761

Pengujian pada tingkat beban tinggi menunjukkan waktu respons API yang lebih bervariasi dan umumnya lebih tinggi dibandingkan dengan tingkat beban normal. Endpoint "/api/table" memiliki waktu respons rata-rata tertinggi dengan 5872 ms, yang merupakan indikasi jelas adanya bottleneck atau proses yang sangat berat pada endpoint ini. Selain itu, endpoint seperti "/api/reservasi" dan "/api/input" juga menunjukkan waktu respons yang tinggi, dengan nilai maksimum mencapai 8018 ms pada "/api/count". Di sisi lain, beberapa endpoint seperti "/api/allsoap" dan "/api/soap" memiliki waktu respons yang lebih rendah, dengan rata-rata di bawah 3000 ms. Secara keseluruhan, meskipun beberapa endpoint menunjukkan waktu respons yang sangat tinggi, waktu respons rata-rata untuk semua endpoint adalah 3289 ms, yang menunjukkan bahwa sebagian besar endpoint masih memiliki kinerja yang dapat diterima pada tingkat beban tinggi, namun diperlukan perhatian lebih lanjut untuk mengoptimalkan endpoint dengan waktu respons tinggi.

TABEL 3 Tingkatan Beban Tinggi

Endpoint	Average (ms)	Min (ms)	Maximum (ms)
/api/allsoap	2289	650	4095
/api/bidanlogin	3153	609	5016
/api/chart	2674	287	4012
/api/count	3007	299	4988
/api/countanually	3624	283	4037
/api/deletebidan	2587	276	4053
/api/edit	2953	283	4039
/api/export	3429	417	5005
/api/findpasien	3062	369	5004
/api/getbidan	3588	626	4993
/api/getreservasi	3367	203	5011
/api/input	3876	847	5993
/api/registbidan	2816	876	3980
/api/reservasi	4090	1045	6010
/api/soap	2237	998	3042
/api/table	5872	1053	8039
Rata-rata	3289	570	4831

C. Uji Keamanan Sistem

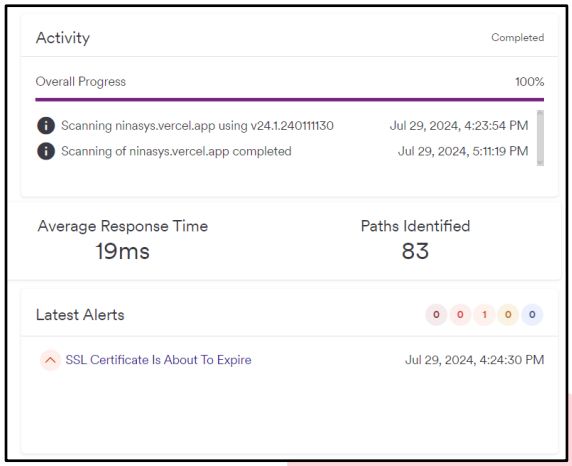
Pengujian dilakukan menggunakan alat uji keamanan Acunetix Security Audit untuk memeriksa kerentanan dalam sistem. Hasil dari pengujian ini memberikan gambaran tentang tingkat ancaman yang ada, serta memberikan rekomendasi perbaikan untuk mengurangi risiko.Seluruh proses pengujian dilakukan dengan menggunakan profil “Critical / High Risk”, yang mencakup berbagai jenis kerentanan seperti SSL/TLS, konfigurasi server, injeksi kode, dan lainnya.

Setelah dilakukan pengujian, terdapat 1 kerentanan yang ditemukan hasil dari pengujian kemanan sistem dengan profil “*Critical / High Risk*”. Selama pengujian, ditemukan bahwa salah satu sertifikat TLS/SSL yang digunakan oleh server hampir kedaluwarsa, yang merupakan kerentanan dengan tingkat ancaman sedang.

Sertifikat yang hampir kedaluwarsa ini memiliki nomor seri 0391c662a89c473a73aac6f2d12d222379a9. Masa berlaku sertifikat ini adalah dari tanggal 14 Juni 2024 pukul 19:57:45 GMT+0700 hingga 12 September 2024 pukul 19:57:44 GMT+0700. Ketika sertifikat tersebut kedaluwarsa, sebagian besar browser web akan menampilkan peringatan keamanan kepada pengguna, meminta mereka untuk secara manual mengonfirmasi keaslian sertifikat.

Hasil pengujian menunjukkan bahwa sistem dalam kondisi yang relatif aman dengan hanya satu kerentanan yang ditemukan, yaitu sertifikat SSL yang hampir kedaluwarsa. Ini merupakan temuan yang penting namun tidak kritikal, karena masih ada waktu untuk memperbarui sertifikat sebelum kedaluwarsa. Tidak ada kerentanan kritikal lainnya yang ditemukan selama pengujian, yang menunjukkan bahwa sebagian besar aspek keamanan telah diterapkan dengan baik.

Meskipun demikian, penting untuk segera memperbarui sertifikat SSL untuk mencegah potensi serangan MITM.



GAMBAR 9 Uji Keamanan Sistem

D. Pengembangan Lebih Lanjut

Pada proyek ini, masih terdapat berbagai fitur yang memiliki potensi untuk dikembangkan lebih lanjut. Fitur-fitur tersebut mencakup peningkatan akses agar bidan dan superadmin dapat tetap menggunakan aplikasi rekam medis tanpa perlu terhubung ke internet, penambahan layanan rekam medis yang mencakup layanan untuk ibu dan anak, serta penambahan fitur pengingat otomatis yang dapat terintegrasi dengan aplikasi sosial media seperti WhatsApp.

V. KESIMPULAN

Aplikasi Rekam Medis Elektronik dengan sistem yang tepat memungkinkan akses cepat dan akurat ke catatan medis pasien. Aplikasi ini dapat diakses melalui perangkat digital seperti komputer atau laptop, sehingga informasi pasien dapat diambil dalam waktu singkat. Menggunakan rekam medis elektronik dan menyimpan data di cloud atau server yang aman juga dapat mengurangi risiko kehilangan atau kerusakan catatan medis fisik. Selain itu, sistem pencatatan yang efisien dan terstruktur dapat diimplementasikan dalam aplikasi rekam medis dengan fitur-fitur seperti formulir untuk setiap layanan, pengingat tindak lanjut layanan, serta kemampuan untuk mengelompokkan dan mengkategorikan informasi medis. Sistem ini juga dapat menyertakan alat analitik untuk membantu bidan dalam menganalisis data kesehatan pasien secara lebih mendalam.

Di sisi lain, aplikasi reservasi pasien terbukti memudahkan pasien dalam menjadwalkan kunjungan. Hal ini dapat dilihat dari persentase dalam pengujian beta yang menunjukkan tingkat kepuasan pengguna sebesar 84,2%. Selain itu, menyediakan informasi lengkap mengenai layanan kesehatan, jam operasional, dan cara mengakses klinik melalui situs web dapat membantu pasien merasa lebih terinformasi dan nyaman dalam melakukan reservasi.

REFERENSI

[1] Y. Y. Putranto, T. W. Adi Putra, and F. N. Hakim, “RANCANG BANGUN SISTEM INFORMASI REKAM MEDIS KLINIK BERBASIS WEB (STUDI KASUS: KLINIK UTAMA MEDITAMA SEMARANG),” Jurnal Informatika Upgris, vol. 3, no. 2, Dec. 2017, doi: 10.26877/jiu.v3i2.1825.

[2] E. wilda Faida, “Analisis Kesiapan Rekam Medik Elektronik Dengan Metode Technology Readiness Index Rumah Sakit Universitas Airlangga Surabaya,” Jurnal Kesehatan, vol. 7, no. 3, pp. 140–154, Jan. 2020, doi: 10.25047/j-kes.v7i3.121.

[3] S. K. Bell et al., “Frequency and Types of Patient-Reported Errors in Electronic Health Record Ambulatory Care Notes,” JAMA Netw Open, vol. 3, no. 6, p. e205867, Jun. 2020, doi: 10.1001/jamanetworkopen.2020.5867.

[4] S. Kom. M. M. M. Kom. W. Suadi and S. Kom. M. Kom. B. A. Pratomo, “IMPLEMENTASI DATABASE ABSTRACTION LAYER UNTUK MYSQL MENGGUNAKAN GOOGLE GO,” 2011. [Online]. Available: https://api.semanticscholar.org/CorpusID:61361964

[5] M. D. Lusita, H. Hurnianingsih, and E. Rihyanti, “Aplikasi Bot Akademik BAAK STMIK Jakarta STI&K Platform Line Messenger Menggunakan Go Languages,” Jurnal Teknologi Sistem Informasi dan Aplikasi, vol. 3, no. 1, p. 1, Feb. 2020, doi: 10.32493/jtsi.v3i1.4130.

[6] W. Ananda, M. Arif, F. Ridha, and Y. Fitrisia, “Pengembangan Cloud Computing Platform As A Service Untuk Bahasa Pemrograman Go,” Aug. 2016.

[7] D. Hows, P. Membrey, E. Plugge, and T. Hawkins, The Definitive Guide to MongoDB. Berkeley, CA: Apress, 2013. doi: 10.1007/978-1-4302-5822-3.

[8] G. Harrison and M. Harrison, MongoDB Performance Tuning. Berkeley, CA: Apress, 2021. doi: 10.1007/978-1-4842-6879-7.

[9] A. Giamas, Mastering MongoDB 4.x, 2nd ed. Birmingham, England: Packt Publishing, 2019.

[10] Kyle, MongoDB in Action. New York, NY: Manning Publications, 2014.