

**RANCANG BANGUN SISTEM INFORMASI MANAJEMEN  
KLINIK GIGI**

(Studi Kasus : Klinik Gigi Xenon Dental House)

**TUGAS AKHIR**

Diajukan Sebagai Salah Satu Syarat untuk Menyelesaikan Program Strata-1 pada  
Departemen Sistem Informasi Fakultas Teknologi Informasi Universitas Andalas

**Oleh :**

**Alif Abdul Rauf**

**2011522024**

**Pembimbing:**

**Husnil Kamil, MT**

**198201182008121002**



**DEPARTEMEN SISTEM INFORMASI  
FAKULTAS TEKNOLOGI INFORMASI  
UNIVERSITAS ANDALAS  
2024**

## **PERNYATAAN**

Saya menyatakan bahwa laporan tugas akhir berjudul "**Rancang Bangun Sistem Informasi Manajemen Klinik Gigi (Studi Kasus: Klinik Gigi Xenon Dental House)**" ini merupakan karya asli saya dan sepanjang pengetahuan saya tidak ada karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain untuk memenuhi syarat tugas akhir di perguruan tinggi lain, kecuali yang secara tertulis diacu dalam naskah ini dan dicantumkan dalam daftar pustaka.

Padang, 30 Agustus 2024

Penulis



Alif Abdul Rauf

## **KATA PENGANTAR**

Puji syukur kehadirat Allah SWT atas rahmat dan karunia-Nya sehingga saya dapat menyelesaikan Laporan Tugas Akhir yang berjudul “Rancang Bangun Sistem Informasi Manajemen Klinik Gigi (Studi Kasus: Klinik Gigi Xenon Dental House)” sebagai salah satu syarat akademik untuk menyelesaikan mata kuliah tugas akhir di Departemen Sistem Informasi, Fakultas Teknologi Informasi, Universitas Andalas.

Dalam proses penyusunan tugas akhir ini, saya menerima banyak bantuan dan dukungan dari berbagai pihak. Oleh karena itu, saya ingin menyampaikan rasa terima kasih yang sebesar-besarnya kepada:

1. Kedua orang tua dan keluarga yang selalu memberikan nasihat, semangat, dan doa dalam proses penyusunan tugas akhir ini.
2. Bapak Husnil Kamil, M.T., selaku Ketua Program Studi Sistem Informasi Universitas Andalas sekaligus pembimbing tugas akhir yang telah memberikan bimbingan selama proses penyusunan tugas akhir ini.
3. Bapak Fajril Akbar, M.Sc., selaku dosen pembimbing akademik yang senantiasa memberikan semangat dan arahan selama masa studi di Sistem Informasi Universitas Andalas.
4. Teman-teman yang telah membantu dan mendampingi saya selama masa studi di Universitas Andalas yang tidak dapat disebutkan satu per satu.
5. Klinik Gigi Xenon Dental House serta seluruh pihak yang telah membantu dan memberikan dukungan dalam penyusunan tugas akhir ini.

Penulis menyadari bahwa laporan tugas akhir ini masih jauh dari sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun dari para pembaca. Kritik dan saran tersebut dapat disampaikan melalui email: alifabdulrauf@gmail.com. Semoga laporan tugas akhir ini dapat bermanfaat bagi penulis dan para pembaca.

Padang, 30 Agustus 2024

Penulis,



Alif Abdul Rauf

## DAFTAR ISI

PERNYATAAN.....	ii
KATA PENGANTAR .....	iii
DAFTAR ISI.....	v
DAFTAR GAMBAR .....	vii
DAFTAR TABEL.....	x
ABSTRAK .....	xi
BAB I PENDAHULUAN.....	1
1.1    Latar Belakang.....	1
1.2    Rumusan Masalah .....	4
1.3    Batasan Masalah.....	4
1.4    Tujuan Penelitian.....	4
1.5    Sistematika Penulisan.....	4
BAB II TINJAUAN PUSTAKA.....	6
2.1    Klinik Gigi.....	6
2.2    Klinik Gigi Xenon Dental House .....	8
2.3    Sistem Informasi Manajemen.....	9
2.4    Progressive Web Apps (PWA) .....	10
2.5    Alat Analisis dan Perancangan .....	11
2.5.1    Business Process Modelling Notation (BPMN) .....	11
2.5.2    Use Case Diagram .....	14
2.5.3    Entity Relationship Diagram(ERD).....	16
BAB III METODE PENELITIAN.....	18
3.1    Objek Penelitian .....	18
3.2    Metode Pengumpulan Data .....	19
3.2.1    Observasi .....	19
3.2.2    Wawancara.....	19
3.2.3    Analisis Dokumen.....	19
3.2.4    Studi Literatur.....	20
3.3    Metode Pengembangan .....	20
3.4    Flowchart Penelitian.....	22
BAB IV ANALISIS DAN PERANCANGAN .....	24
4.1    Analisis Sistem .....	24

4.1.1	Analisis sistem yang Sedang Berjalan .....	24
4.1.2	Sistem yang Diusulkan .....	27
4.1.3	Analisis Kebutuhan Fungsional .....	30
4.1.4	Use Case Diagram .....	32
4.1.5	Sequence Diagram .....	34
4.2	Perancangan sistem .....	37
4.2.1	Perancangan Basis data.....	37
4.2.2	Arsitektur Aplikasi.....	43
4.2.3	Class Diagram.....	45
4.2.4	Tampilan Mockup Antarmuka.....	49
<b>BAB V IMPLEMENTASI DAN PENGUJIAN .....</b>		<b>54</b>
5.1	Implementasi Sistem .....	54
5.1.1	Pengkodean sistem.....	54
5.1.2	Implementasi Antarmuka sistem .....	65
5.2	Pengujian Sistem .....	68
5.2.1	Fokus Pengujian.....	68
5.2.2	Kasus Hasil Pengujian .....	70
5.2.3	Kesimpulan hasil pengujian.....	82
5.2.4	Analisa dan Pembahasan Hasil .....	85
<b>BAB VI KESIMPULAN DAN SARAN .....</b>		<b>89</b>
6.1	Kesimpulan.....	89
6.2	Saran .....	89
<b>DAFTAR PUSTAKA .....</b>		<b>90</b>
<b>LAMPIRAN A .....</b>		<b>93</b>
<b>LAMPIRAN B .....</b>		<b>105</b>
<b>LAMPIRAN C .....</b>		<b>109</b>
<b>LAMPIRAN D .....</b>		<b>125</b>
<b>LAMPIRAN E .....</b>		<b>145</b>
<b>LAMPIRAN F .....</b>		<b>197</b>
<b>LAMPIRAN G .....</b>		<b>204</b>

## **DAFTAR GAMBAR**

Gambar 2.1 Struktur Organisasi Klinik Gigi Xenon Dental House .....	8
Gambar 2.2 Penerapan PWA pada <i>website</i> trivago(Warrender, 2018) .....	10
Gambar 2. 3 Notasi Swimlanes (Ismanto, Hidayah and Charisma, 2020) ....	12
Gambar 2. 4 Notasi Connecting Object (Ismanto, Hidayah and Charisma, 2020).....	13
Gambar 2. 5 Notasi Start, Intermediate, dan End Event(Wagner, 2017) .....	13
Gambar 2. 6 <i>Manual, user, service, receive task</i> (Stiehl, Raw and Smith, 2014).....	13
Gambar 2. 7 Notasi Gateway(Stiehl, Raw and Smith, 2014) .....	14
Gambar 2. 8 Notasi Artifacts (Ismanto, Hidayah and Charisma, 2020).....	14
Gambar 2.9 ERD hubungan instruktur dan pelajar (Silberschatz, 2011) .....	16
Gambar 3.1 Waterfall Model(Sommerville, 2011).....	21
Gambar 3.2 Tahapan Penelitian.....	23
Gambar 4.1 BPMN Sistem reservasi .....	25
Gambar 4.2 BPMN Sistem pelayanan .....	26
Gambar 4.3 BPMN manajemen jadwal .....	27
Gambar 4.4 BPMN Sistem reservasi .....	28
Gambar 4.5 BPMN Sistem pelayanan yang diusulkan.....	29
Gambar 4.6 BPMN Perizinan dokter.....	30
Gambar 4.7 Use Case Diagram Sistem.....	33
Gambar 4.8 Sequence Diagram input data user.....	34
Gambar 4.9 Sequence diagram login.....	35
Gambar 4.10 Sequence diagram reservasi admin.....	36
Gambar 4.11 Sequence diagram lihat dashboard owner.....	36
Gambar 4.12 ERD Sistem Informasi Manajemen Klinik .....	37
Gambar 4.13 Arsitektur Aplikasi <i>Website</i> .....	43
Gambar 4.14 Class Diagram Konfirmasi Reservasi Dari Pasien.....	46
Gambar 4.15 Class Diagram Pengajuan Permintaan Izin Dokter.....	47
Gambar 4.16 Class Diagram Pemilik Memutuskan Perizinan Dokter .....	48
Gambar 4.17 Mockup Form Reservasi .....	49
Gambar 4.18 Mockup Tampilan data user .....	50
Gambar 4.19 Mockup Dashboard akun owner .....	51

Gambar 4.20 Mockup Landing page website .....	52
Gambar 4.20 Mockup Antarmuka Offline PWA.....	53
Gambar 5.1 Potongan Kode Route Dokter .....	55
Gambar 5.2 Potongan Kode Route admin .....	56
Gambar 5.3 Potongan kode DataDokterController.....	57
Gambar 5.4 Potongan kode DataDokterController.....	58
Gambar 5.5 Potongan kode model detail jadwal .....	60
Gambar 5.6 Potongan kode model dokter .....	61
Gambar 5.7 Potongan kode view datadokter_add .....	62
Gambar 5.8 Potongan kode View datadokter .....	63
Gambar 5.9 Kode serviceworker.js.....	64
Gambar 5.10 Kode manifest.json .....	64
Gambar 5.11 Implementasi Antarmuka Halaman Dashboard Owner .....	65
Gambar 5.12 Implementasi Antarmuka Dashboard Admin .....	66
Gambar 5.13 Implementasi Antarmuka Reservasi dari admin .....	67
Gambar 5.14 Implementasi Antarmuka Reservasi Pasien.....	67
Gambar 5.15 Tampilan awal reservasi dari admin .....	71
Gambar 5.16 Tampilan Form tambah reservasi.....	71
Gambar 5.17 Tampilan reservasi setelah data berhasil ditambahkan.....	72
Gambar 5.18 Tampilan notifikasi saat input form reservasi tidak diisi.....	73
Gambar 5.19 Tampilan form registrasi akun .....	74
Gambar 5.20 Tampilan Setelah pasien berhasil register.....	74
Gambar 5.21 Penambahan data user setelah registrasi akun pasien dilakukan .....	75
Gambar 5.22 Notifikasi saat mendaftarkan akun email yang sudah ada .....	76
Gambar 5.23 Tampilan form saat pasien melakukan reservasi .....	77
Gambar 5.24 Tampilan history pasien selesai melakukan reservasi .....	77
Gambar 5.25 Tampilan user admin setelah pasien selesai melakukan reservasi .....	78
Gambar 5.26 Tampilan data reservasi admin setelah reservasi pasien diterima .....	78
Gambar 5.27 Tampilan data reservasi admin setelah reservasi pasien diterima .....	78
Gambar 5.28 Tampilan history reservasi pasien setelah melakukan reservasi .....	79

Gambar 5.29 Tampilan reservasi pasien setelah melakukan reservasi pada admin .....	80
Gambar 5.30 Tampilan histori reservasi pasien setelah ditolak admin .....	80
Gambar 5.31 Tampilan landing page saat server aktif .....	81
Gambar 5.32 Tampilan saat server dimatikan .....	82
Gambar 5.33 Tampilan landing page tetap muncul setelah server mati.....	82

## **DAFTAR TABEL**

Tabel 2.1 Simbol dalam Use Case Diagram.....	15
Tabel 4.1 Tabel Perawatan .....	38
Tabel 4.2 Tabel Lokasi .....	38
Tabel 4.3 Tabel Detail jadwal .....	39
Tabel 4.4 Tabel Pengeluaran.....	39
Tabel 4.5 Tabel Izin Dokter .....	40
Tabel 4.6 Tabel Users .....	40
Tabel 4.7 Tabel Dokter .....	41
Tabel 4.8 Tabel Perawatan_reservasi.....	41
Tabel 4.9 Tabel Perawatan .....	42
Tabel 4.10 Tabel Pasien.....	42
Tabel 5.1 Pengujian Aplikasi .....	68
Tabel 5.2 Pengujian kondisi benar input reservasi dari admin.....	70
Tabel 5.3 Pengujian alternatif reservasi dari admin .....	72
Tabel 5.4 Pengujian kondisi benar registrasi pasien.....	73
Tabel 5.5 Pengujian alternatif registrasi akun pasien.....	75
Tabel 5.6 Pengujian kondisi benar reservasi dari pasien .....	76
Tabel 5.7 Pengujian alternatif reservasi dari pasien .....	79
Tabel 5.8 Pengujian mode offline pada <i>website</i> .....	81
Tabel 5.9 Kesimpulan hasil pengujian .....	83

## **ABSTRAK**

*Klinik Gigi Xenon Dental House adalah klinik gigi yang menyediakan layanan kesehatan gigi yang didirikan pada tahun 2020 di Kota Padang dan telah berkembang dengan memiliki cabang di Kota Bukittinggi. Dalam operasionalnya, klinik ini menghadapi beberapa tantangan dalam pengelolaan data dan layanan. Pertama, proses reservasi masih dilakukan secara manual melalui WhatsApp, yang dapat menyebabkan miskomunikasi dan kesalahan penjadwalan. Kedua, tidak adanya satu pusat data yang terintegrasi yang mana saat ini menggunakan berbagai media untuk mencatat data, mulai dari catatan manual hingga pencatatan reservasi melalui WhatsApp. Ketiga, pencatatan rekam medis masih menggunakan media kertas yang berisiko rusak atau hilang serta membutuhkan ruang penyimpanan fisik dan . Keempat, tidak adanya sistem visualisasi data real-time yang dapat menyulitkan pemantauan kinerja klinik, tren pasien, tren layanan perawatan, dan produktivitas dokter. Oleh karena itu, dibutuhkan sebuah sistem informasi manajemen klinik gigi untuk mengoptimalkan pengelolaan operasional klinik. Penelitian ini menggunakan metode waterfall dengan teknik pengumpulan data melalui observasi, wawancara, analisis dokumen, dan studi literatur. Sistem informasi yang dibangun berbasis web dengan mengimplementasikan teknologi Progressive Web Apps (PWA), menggunakan framework Laravel 11, dan API Twilio untuk integrasi notifikasi WhatsApp. Hasil penelitian menunjukkan bahwa sistem informasi manajemen klinik gigi berhasil dibangun sesuai dengan perancangan. Sistem ini menyediakan sentralisasi data, meningkatkan keamanan dan akurasi data, mengotomatisasi penjadwalan dan reservasi, serta menyediakan fitur visualisasi data untuk pemantauan kinerja klinik. Implementasi PWA memungkinkan akses offline, sementara integrasi WhatsApp memfasilitasi komunikasi antara klinik dan pasien.*

*Kata Kunci:* *Sistem Informasi Manajemen, Klinik Gigi, Progressive Web Apps, Laravel, Twilio, Visualisasi Data*

## **BAB I**

### **PENDAHULUAN**

#### **1.1 Latar Belakang**

Dalam era globalisasi yang semakin maju, perkembangan teknologi informasi menjadi esensial sebagai alat pendukung untuk mempermudah berbagai aktivitas di berbagai sektor, termasuk di dalamnya organisasi, lembaga, instansi, dan perusahaan. Penerapan teknologi informasi telah membawa berbagai kemudahan yang signifikan dalam kehidupan sehari-hari, termasuk di bidang pelayanan kesehatan. Meskipun sistem informasi telah ada dalam berbagai organisasi, termasuk klinik gigi, masih terdapat permasalahan mendasar yang perlu diatasi.(Indah *et al.*, 2021).

Sebagai contoh, kita bisa merujuk pada Klinik Gigi Xenon Dental House, yang merupakan salah satu penyedia layanan kesehatan gigi di Kota Padang. Setelah dilakukan wawancara dan karyawan dengan Pemilik Klinik Gigi Xenon Dental House, proses bisnis pada klinik ini masih melakukan sebagian besar administrasinya secara manual yang mana bisa dikembangkan menjadi proses bisnis yang lebih baik lagi. Pendaftaran pasien, dan proses pencatatan masih bergantung pada proses manual yang sangat bergantung dengan keterampilan dan ketelitian manusia, serta dapat menimbulkan *human error*.

Klinik Gigi Xenon Dental House melayani berbagai kelompok usia, mulai dari anak-anak hingga orang dewasa, yang dilayani oleh dokter-dokter dengan berbagai jadwal reservasi yang ada. Ketidaksesuaian jadwal reservasi terjadi akibat kesalahan dalam penjadwalan yang saat ini dicatat menggunakan aplikasi Whatsapp saja dan tidak menggunakan sistem khusus reservasi. Saat ini reservasi di klinik ini hanya dapat dilakukan dengan menghubungi nomor WhatsApp admin klinik atau datang langsung ke klinik. Selain itu, staf klinik harus membuat laporan rekam medis yang juga dilakukan secara manual menggunakan media kertas yang membutuhkan ruang fisik yang disimpan terpisah dan mudah rusak. Sistem manual yang ada saat ini juga tidak menyediakan kemampuan visualisasi *realtime* data yang mana menyebabkan kurangnya penyajian informasi yang jelas untuk memantau kinerja klinik, tren pasien, tren layanan perawatan, serta produktivitas dokter.

Proses yang manual ini juga membuat ketidakefisienan pada proses klinik gigi, sebagai contoh pada klinik gigi Dental Echo Clinic pengolahan data yang dilakukan secara manual membutuhkan waktu yang lama menimbulkan ketidak validan data dan kerahasiaan data tidak bisa terjaga dengan baik (Lestari, 2019).

Beberapa penelitian sebelumnya telah mengkaji sistem informasi manajemen dalam konteks pelayanan kesehatan. Sebagai contoh, penelitian oleh Mahdalena, Alamsyah & Sidik (2023) yang berjudul “Sistem Informasi Manajemen dan Keuangan Berbasis Web pada Klinik Gigi Eldental Banjarmasin”. Sistem ini bertujuan untuk meningkatkan kualitas pelayanan yang diberikan oleh klinik dengan mengatasi keterbatasan sistem manual yang ada saat ini untuk data pasien, pencatatan keuangan, dan rekam medis. Para peneliti menggunakan metode Waterfall dan bahasa pemrograman PHP dengan Framework Laravel untuk mengembangkan sistem, yang mencakup fitur-fitur seperti manajemen data untuk dokter, perawat, pasien, obat-obatan, keuangan, dan pelaporan. Web ini menghasilkan proses pendataan yang lebih efisien, dan mempermudah klinik dalam pembuatan laporan keuangan dan inventaris.

Penelitian oleh Rafiqah Majidah dkk (Majidah, Rusdianto ,2019) berjudul “Pengembangan Sistem Informasi Pengelolaan Klinik Gigi Berbasis Website Menggunakan Prinsip Point of Sale” juga membahas penerapan sistem informasi pada klinik. Jurnal mengusulkan sebuah sistem yang menggabungkan prinsip-prinsip point of sale dengan sistem berbasis *website* penerapan sistem informasi pengelolaan klinik gigi menggunakan *point of sale*. Selain itu, sistem ini juga membantu mengatasi masalah kurangnya efisiensi kerja dalam hal waktu dan upaya yang dikeluarkan hingga 30 kali lebih cepat dibandingkan dengan sebelum menggunakan sistem.

Penelitian lainnya yang berjudul "Sistem Informasi Manajemen Klinik Gigi Berbasis Client Server (Sari Ira Puspita , 2017). Penelitian ini membahas tentang sistem informasi manajemen klinik gigi yang bertujuan untuk menggantikan sistem pencatatan manual dan pengolahan data ke sistem komputerisasi, untuk mengatasi hambatan yang sering terjadi terkait dengan ketidakmampuan untuk menyediakan informasi secara cepat, akurat, dan tepat waktu. Sistem ini mencakup fitur-fitur seperti pendaftaran pasien, rekam medis, diagnosis, pengobatan, dan resep.

Penelitian ini menghasilkan peningkatan pelayanan pada pihak RSJ Pekanbaru serta memudahkan, memperpendek proses, dan menghemat waktu dalam penggerjaan proses pelayanan klinik gigi.

Penelitian-penelitian ini relevan dengan penelitian yang sedang dilakukan, yang bertujuan untuk mengembangkan sistem informasi manajemen klinik gigi yang lebih baik. Dari penelitian penelitian ini juga terlihat bahwa penerapan sistem informasi pada klinik dibutuhkan dan dapat meningkatkan efisiensi dan kualitas dari dari pelayanan klinik. Meskipun terdapat penelitian terkait, belum ada yang menerapkan pendekatan Progressive Web Applications (PWA). PWA merupakan teknologi Web-based development, yang memungkinkan agar suatu *website* dapat memiliki karakteristik / *experience* layaknya menggunakan suatu aplikasi mobile native, sehingga dapat memberikan pengalaman yang berkesan kepada user(Karli, Muawwal and Thayf, 2023). Dalam penelitian ini, sistem informasi yang dirancang akan menggunakan pendekatan Progressive Web Applications (PWA) yang membuat *website* dapat dijalankan dalam keadaan *offline* dan lebih cepat dalam memuat data(Muddin, Tehuayo and Iksan, 2021). Sistem informasi ini semoga memberikan kemudahan kepada pasien dan membantu klinik dalam pendaftaran pasien, pengelolaan data yang, pelayanan yang lebih baik, peningkatan produktivitas, dan proses yang lebih baik. Sistem informasi ini diharapkan dapat diakses dan dimanfaatkan oleh masyarakat luas, sehingga dapat berfungsi secara optimal di Klinik Gigi Xenon Dental House.

Berdasarkan dari permasalahan tersebut dilaksanakan penelitian di Klinik Gigi Xenon Dental House dengan judul “RANCANG BANGUN SISTEM INFORMASI MANAJEMEN KLINIK GIGI” (Studi Kasus : Klinik Gigi Xenon Dental House)”. Dengan penelitian ini, diharapkan dapat memberikan kontribusi yang berarti terhadap klinik ini dan pada akhirnya meningkatkan kualitas layanan kesehatan gigi di masyarakat.

## **1.2 Rumusan Masalah**

Berdasarkan latar belakang yang telah disebutkan sebelumnya, rumusan masalah dalam penelitian ini adalah bagaimana membangun sistem informasi manajemen klinik gigi pada klinik gigi Xenon Dental House.

## **1.3 Batasan Masalah**

Berdasarkan rumusan masalah seperti yang dijelaskan sebelumnya, maka batasan masalah dalam tugas akhir ini adalah:

1. Sistem informasi ini mencakup pengelolaan data pasien, dokter, pengelolaan data rekam medis, pengelolaan reservasi, izin dokter, pengelolaan data layanan perawatan, visualisasi data klinik.
2. Sistem informasi manajemen pada klinik gigi xenon dental house dilakukan sampai tahap implementasi dan pengujian.
3. Pengujian aplikasi hanya sebatas memeriksa ketersediaan fungsional dan kesesuaian dengan rancangan sistem yang diusulkan.

## **1.4 Tujuan Penelitian**

Tujuan dari penelitian ini adalah membangun sebuah sistem informasi manajemen klinik gigi pada Klinik Gigi Xenon Dental House untuk mengatasi kendala serta membantu pada manajemen klinik gigi seperti pengelolaan data pasien, dokter, pengelolaan data rekam medis, pengelolaan reservasi, izin dokter, pengelolaan data pelayanan perawatan, visualisasi data klinik di Klinik Gigi Xenon Dental House.

## **1.5 Sistematika Penulisan**

Sistematika penulisan penelitian ini adalah sebagai berikut:

### **BAB I : PENDAHULUAN**

Bab ini berisi tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, serta sistematika penulisan laporan.

### **BAB II : TINJAUAN PUSTAKA**

Bab ini berisi tentang landasan teori dan informasi pendukung yang digunakan untuk penelitian ini.

### **BAB III : METODOLOGI PENELITIAN**

Bab ini menjelaskan tentang objek penelitian, metode pengumpulan data, metode pengembangan sistem yang digunakan, dan flowchart penelitian.

### **BAB IV: ANALISIS DAN PERANCANGAN SISTEM**

Bab ini berisi tentang analisis sistem yang berjalan, analisis kebutuhan dan perancangan pada sistem usulan untuk menjawab permasalahan pada sistem lama yang digambarkan melalui diagram dan tools pendukung lainnya.

### **BAB V: IMPLEMENTASI DAN PENGUJIAN**

Bab ini berisi tentang implementasi dari hasil analisis dan perancangan sistem yang telah dijabarkan pada bab sebelumnya. Pada bab ini juga akan dijelaskan pengujian yang dilakukan terhadap sistem usulan berdasarkan rancangan yang telah dibuat pada bab sebelumnya.

### **BAB VI: PENUTUP**

Bab ini berisi tentang kesimpulan dari penelitian dan saran terhadap pengembangan kedepannya.

## **BAB II**

### **TINJAUAN PUSTAKA**

Bab ini menjelaskan teori dan informasi pendukung yang digunakan dalam penelitian pembangunan sistem informasi manajemen klinik gigi xenon dental house.

#### **2.1 Klinik Gigi**

Dalam Peraturan Menteri Kesehatan nomor 028/Menkes/Per/I/2011 (Kemenkes RI 2011, 2011), klinik dijelaskan sebagai fasilitas pelayanan kesehatan yang menyelenggarakan pelayanan kesehatan perorangan yang menyediakan pelayanan medis dasar dan atau spesialistik, diselenggarakan oleh lebih dari satu jenis tenaga kesehatan dan dipimpin oleh seorang tenaga medis.. Lebih khusus, klinik gigi adalah klinik yang memberikan pelayanan medik dasar yang dapat mengkhususkan pelayanan pada satu bidang tertentu berdasarkan disiplin ilmu, organ atau jenis penyakit tertentu(Kemenkes RI 2011, 2011) .

Biasanya, ketika seseorang datang ke klinik atau praktik dokter gigi pribadi, tidak disebutkan secara eksplisit kategori atau jenis klinik gigi yang dikunjungi, karena klinik gigi yang dikunjungi cenderung menyediakan beragam jenis perawatan gigi dan mulut. Namun, dalam konteks rumah sakit atau poliklinik gigi, ruang perawatan gigi sering kali dibagi menjadi berbagai kategori sesuai dengan jenis perawatan yang disediakan oleh dokter gigi yang bertugas di klinik tersebut. Setiap jenis klinik gigi umumnya memiliki dokter gigi spesialis yang berkompeten di bidangnya(Muntihana *et al.*, 2017).

Klinik tidak hanya berfungsi sebagai tempat untuk memberikan layanan medis, tetapi juga memiliki peran penting dalam membantu masyarakat sekitarnya melalui pengamatan terhadap kondisi tubuh mereka. Pengamatan ini dapat dilakukan melalui pemeriksaan terhadap keluhan-keluhan yang disampaikan oleh pasien. Seluruh hasil pemeriksaan dokter kemudian terdokumentasikan dalam rekam medik, mencakup diagnosa penyakit, tindakan medis, dan resep obat. Proses di klinik, selain mencakup pemeriksaan kondisi tubuh pasien, juga melibatkan

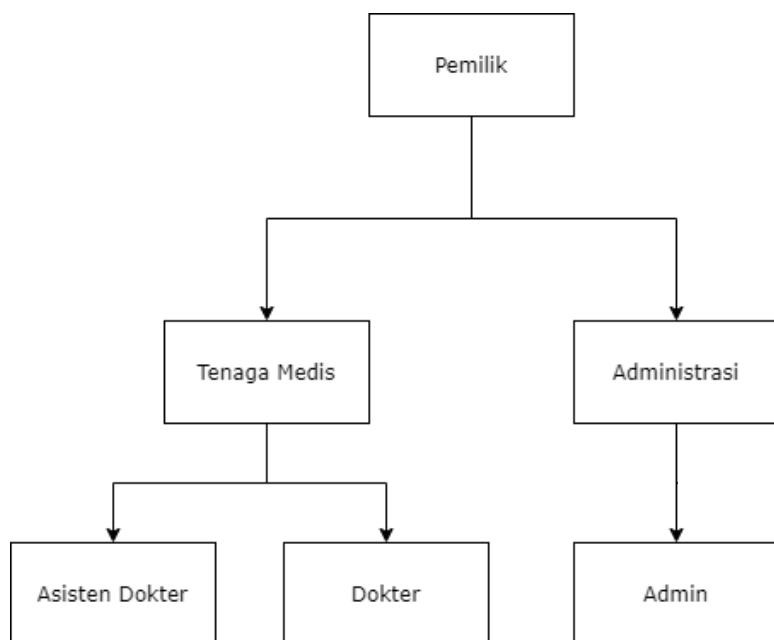
administrasi klinik yang berfungsi untuk mencatat semua kegiatan sesuai dengan proses bisnis yang berlangsung. Hal ini mencakup pendaftaran pasien, pencatatan data obat, pemeriksaan laboratorium, dan pembayaran obat. Pentingnya informasi dalam konteks pelayanan kesehatan, termasuk di dalamnya klinik, sangat besar. Informasi yang diperoleh dengan cepat, tepat, dan akurat akan berkontribusi pada penyelenggaraan layanan yang terbaik bagi pasien, meningkatkan tingkat kepuasan mereka. Penggunaan komputer dalam pengolahan data di klinik akan membantu meningkatkan efektivitas dan efisiensi proses tersebut.

Klinik, sebagai fasilitas pelayanan kesehatan, memiliki peran signifikan tidak hanya dalam merawat pasien, tetapi juga dalam membantu masyarakat sekitar melalui observasi terhadap kondisi tubuh mereka. Observasi ini dilakukan melalui pemeriksaan terhadap keluhan-keluhan pasien, di mana hasilnya dicatat dalam rekam medik yang mencakup diagnosa penyakit, tindakan dokter, dan resep obat. Meskipun fokus utama adalah pada pemeriksaan kondisi kesehatan pasien, klinik juga melibatkan administrasi yang mendukung proses bisnisnya. Administrasi klinik mencakup kegiatan seperti pendaftaran pasien, pencatatan data obat, pemeriksaan laboratorium, dan pembayaran obat. Pentingnya informasi dalam pelayanan kesehatan seperti informasi yang cepat, tepat, dan valid akan memberikan kepuasan pasien.

Penting untuk diingat bahwa klinik gigi tidak hanya berperan sebagai penyedia layanan medis, tetapi juga sebagai entitas yang mendukung kebutuhan administratif dan informasional. Penggunaan teknologi, seperti pengolahan data melalui komputer, di klinik dapat meningkatkan efektivitas dan efisiensi proses tersebut. Informasi yang diperoleh dengan cepat, tepat, dan akurat tidak hanya memperbaiki penyelenggaraan layanan bagi pasien, tetapi juga membantu dalam manajemen administratif, termasuk perekaman data pasien, pengelolaan inventaris obat, dan pelaporan hasil pemeriksaan laboratorium. Dengan demikian, klinik gigi bukan hanya tempat untuk merawat kesehatan gigi masyarakat, tetapi juga menjadi tempat untuk menjaga dan meningkatkan kesehatan serta kepuasan pasien (Nuryani, 2021).

## 2.2 Klinik Gigi Xenon Dental House

Klinik gigi Xenon Dental House, adalah klinik yang berlokasi di Jl. Palembang No 11, Ulak Karang, Kota Padang, merupakan sebuah pusat pelayanan kesehatan gigi yang didirikan pada tahun 2020 melalui kolaborasi beberapa dokter gigi. Saat ini klinik gigi Xenon Dental House sudah terdapat di 2 kota yaitu kota padang dan kota bukittinggi. Dalam jalannya klinik ini tentunya terdapat elemen-elemen yang mendukung klinik gigi Xenon Dental House sebagai organisasi, adapun struktur organisasi dari gigi Xenon Dental House dapat dilihat pada gambar 2.1:



Gambar 2.1 Struktur Organisasi Klinik Gigi Xenon Dental House

Dari diagram diatas, pada klinik gigi Xenon Dental House terdapat beberapa peran yang bertanggung jawab. Pemilik bertanggung jawab atas manajemen keseluruhan dan pengambilan keputusan strategis. Dokter gigi adalah tenaga medis yang memberikan pelayanan kesehatan gigi kepada pasien, sementara asisten dokter mendukung tugas dokter dengan membantu dalam prosedur klinis. Administrasi klinik bertanggung jawab mengelola aspek administratif klinik,

seperti pengelolaan jadwal, sosial media, reservasi. Sementara itu, resepsionis berperan sebagai penerima tamu, menjawab panggilan.

Klinik ini awalnya berfokus pada penyediaan peralatan dan keperluan perawatan gigi. Namun, seiring perkembangannya, Klinik gigi Xenon Dental House mulai menyediakan berbagai layanan kesehatan gigi. Proses pelayanan di klinik ini melibatkan langkah-langkah berikut: pasien melakukan reservasi dengan mengunjungi langsung atau menghubungi melalui aplikasi WhatsApp milik Klinik gigi Xenon Dental House. Data identitas pasien dan keluhan pasien dicatat, lalu pasien diberikan jadwal untuk tindakan dokter. Sehari sebelum jadwal tindakan, admin klinik mengirimkan pengingat kepada pasien. Saat pasien datang, tindakan medis dilakukan sesuai dengan penilaian dokter. Pasien diminta untuk menjadwalkan kunjungan kontrol jika diperlukan. Klinik ini menyediakan berbagai jenis tindakan, termasuk konsultasi dokter, tambal gigi, pencabutan gigi, scaling gigi, dan pemasangan behel.

### **2.3 Sistem Informasi Manajemen**

Manajemen sendiri mencakup proses perencanaan, pengorganisasian, pengawasan, pengarahan, dan lain-lain, dalam suatu organisasi. Sedangkan, informasi dalam satu organisasi adalah data yang diolah sedemikian rupa sehingga memiliki nilai dan arti bagi organisasi. Sistem Informasi Manajemen (SIM) merupakan sistem yang mengolah serta mengorganisasikan data dan informasi yang berguna untuk mendukung pelaksanaan tugas dalam suatu organisasi (Hariyanto, 2018).

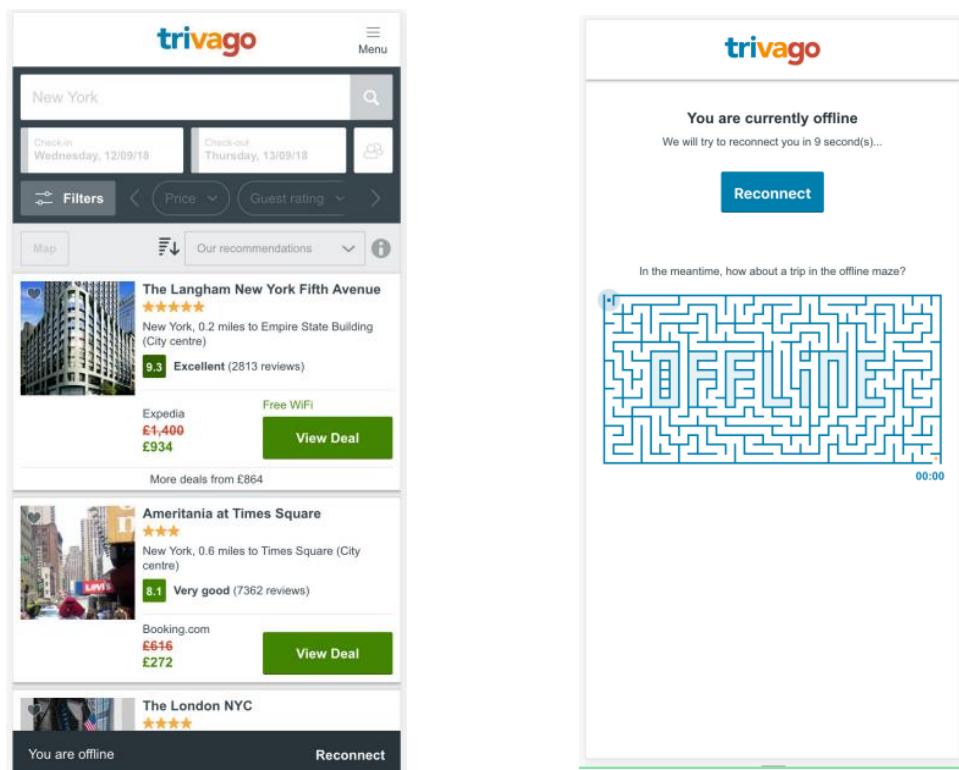
Sistem Informasi Manajemen dapat berupa sistem informasi berikut ini (Wijoyo, 2021) :

- 1) Sistem Informasi Akuntansi
- 2) Sistem Informasi Pemasaran
- 3) Sistem Informasi Manajemen Persediaan.
- 4) Sistem Informasi Personalia.
- 5) Sistem Informasi Distribusi.

- 6) Sistem Informasi Pembelian.
- 7) Sistem Informasi Analisa Kredit.
- 8) Sistem Informasi Analisa Software.
- 9) Sistem Informasi riset dan pengembangan.
- 10) Sistem Informasi kekayaan.
- 11) Sistem Informasi Teknis.

## 2.4 Progressive Web Apps (PWA)

Progressive Web App adalah suatu teknik bagaimana kita dapat mengakses dengan kencang dan cepat di *website* dan aplikasi, aplikasi bisa ditampilkan dalam bentuk *website* ataupun aplikasi. Progressive Web App (PWA) adalah sebuah konsep pengembangan web yang bertujuan untuk memberikan pengalaman yang mirip dengan aplikasi mobile pada peramban (browser) desktop dan perangkat mobile. PWA memanfaatkan kemampuan peramban modern, seperti Service Workers dan Web App Manifest, untuk menciptakan aplikasi web yang lebih kuat dan dapat diakses secara offline (Haryanto and Saputra Elsi, 2021). Contoh penerapan PWA bisa dilihat pada gambar ini Gambar 2.2 berikut ini.



Gambar 2.2 Penerapan PWA pada *website* trivago(Warrender, 2018)

PWA memiliki beberapa manfaat seperti berikut (Aleksandrs Hodakovskis,2019) :

Sebagai pemilik situs web atau eCommerce:

- Memberikan pengalaman website yang responsif
- Performa lebih cepat
- Meningkatkan pengalaman pengguna
- Meningkatkan tingkat pencarian website
- PWA biasanya lebih terjangkau untuk dibuat dan dipelihara dibandingkan aplikasi native
- User dapat menyimpan aplikasi di layar utama mereka
- Semua hal di atas menghasilkan tingkat kinerja yang lebih tinggi secara keseluruhan keterlibatan, yang mengarah pada peningkatan pendapatan

Sebagai pengguna:

- Waktu muat lebih cepat dan instan
- Penjelajahan(searching) offline
- Pengalaman pencarian yang lebih baik dan lancar
- PWA menggunakan lebih sedikit data
- Akses sekali klik (bila disimpan ke layar beranda perangkat Anda)

## 2.5 Alat Analisis dan Perancangan

Dalam bagian ini, diberikan penjelasan mengenai teori-teori terkait dengan *tools* analisis dan perancangan, seperti Business Process Modelling Notation (BPMN), Use Case, dan Entity Relationship Diagram (ERD). Alat-alat ini memiliki hubungan penting dalam proses analisis dan perancangan yang terjadi dalam konteks penelitian ini.

### 2.5.1 Business Process Modelling Notation (BPMN)

*Business Process Modelling Notation* adalah sebuah standar yang digunakan untuk memodelkan proses bisnis dengan menyediakan notasi grafis dalam pemodelan proses bisnis tersebut. BPMN menjelaskan diagram proses bisnis

yang disusun untuk membuat model grafis dari proses bisnis dengan aktivitas dan kontrol aliran yang mendefinisikan urutan kerja berdasarkan pendekatan diagram alur (Yohana and Marisa, 2018).

*Business Process Modelling Notation* terdiri atas empat kategori elemen (Ismanto, Hidayah and Charisma, 2020), yaitu sebagai berikut.

### 1. *Swimlanes*

Elemen ini digunakan untuk mengatur dan memisahkan peran atau tanggung jawab dari suatu proses. Notasi yang digunakan adalah *pool* dan *lane*. *Pool* adalah kontainer dari satu proses. Sedangkan *lane* adalah partisi dari suatu proses, yang menunjukkan sub organisasi, jabatan, peran atau penanggungjawab. Notasi *swimlanes* dapat digambarkan sesuai Gambar 2.3.



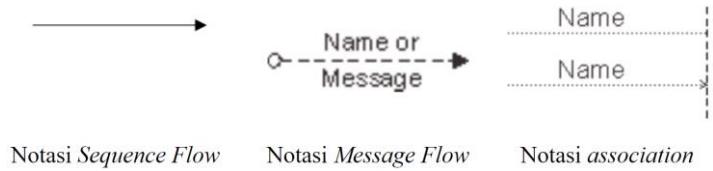
Gambar 2. 3 Notasi Swimlanes (Ismanto, Hidayah and Charisma, 2020)

### 2. *Connecting Object*

Elemen ini digunakan untuk menggambarkan aliran pesan antar proses dimana satu kejadian dengan kejadian yang lain saling berhubungan dan merepresentasikan dari hubungan tersebut. Terdapat tiga notasi yang digunakan pada *connecting object*:

- a. *Sequence flow*, konektor yang menghubungkan antar objek yang mengalir dalam satu proses (*pool*).
- b. *Message flow*, konektor yang menghubungkan antar objek yang mengalir antar proses (beda *pool*).
- c. *Association*, konektor yang menghubungkan objek yang mengalir ke *artifact*.

Masing-masing notasi *connecting object* dapat digambarkan seperti Gambar 2.4.

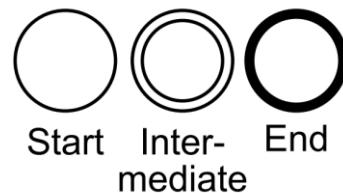


Gambar 2. 4 Notasi Connecting Object (Ismanto, Hidayah and Charisma, 2020)

### 3. Flow Object

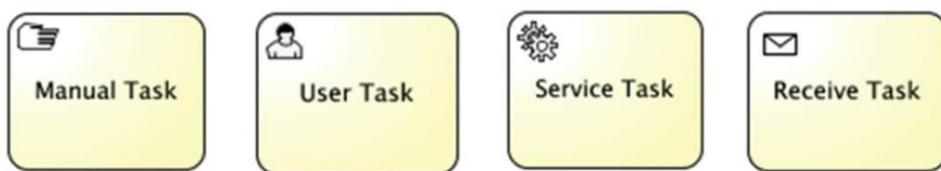
Elemen ini digunakan untuk menunjukkan objek yang mengalir pada suatu proses. Terdapat tiga notasi yang digunakan pada *flow object*:

a. *Event*, menjelaskan apa yang terjadi saat itu. *Event-event* ini mempengaruhi alur proses dan biasanya menyebabkan terjadinya kejadian (*trigger*) atau sebuah dampak (*result*). Ada tiga macam *event*, yaitu *start event*, *intermediate event*, dan *end event* yang dapat dinotasikan sesuai Gambar 2.5.



Gambar 2. 5 Notasi Start, Intermediate, dan End Event(Wagner, 2017)

b. *Activity*, merepresentasikan pekerjaan (*task*) dan subproses(serangkaian pekerjaan) yang harus diselesaikan. Pekerjaan memiliki beberapa kategorikan seperti pada Gambar 2.6 berikut.



Gambar 2. 6 *Manual, user, service, receive task* (Stiehl, Raw and Smith, 2014)

c. *Gateway*, digunakan untuk mengontrol divergensi dan konvergensi *sequential flow*. Notasi *Gateway* dapat dilihat pada Gambar 2.7.



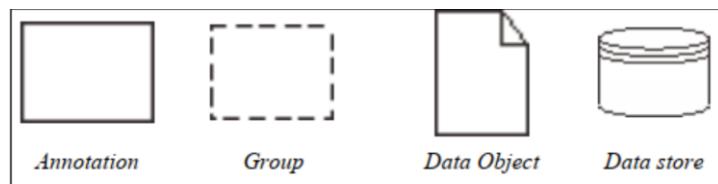
Gambar 2. 7 Notasi Gateway(Stiehl, Raw and Smith, 2014)

#### 4. *Artifacts*

Elemen yang digunakan untuk informasi tambahan pada suatu proses. Terdapat empat notasi yang digunakan pada *artifacts*:

- Annotation*, penjelasan dari suatu objek yang mengalir.
- Group*, pengelompokan dari beberapa objek yang mengalir.
- Data object*, file dan dokumen yang digunakan dan dihasilkan oleh suatu aktifitas.
- Data store*, sistem dan aplikasi yang digunakan dan dihasilkan oleh suatu aktifitas.

Notasi *artifacts* dapat digambarkan seperti Gambar 2.8.



Gambar 2. 8 Notasi Artifacts (Ismanto, Hidayah and Charisma, 2020)

#### 2.5.2 Use Case Diagram

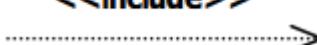
Diagram Use Case adalah representasi visual yang menggambarkan fungsi yang diinginkan dari suatu sistem, dengan penekanan pada apa yang dilakukan oleh

sistem, bukan bagaimana melakukannya. Setiap use case menggambarkan interaksi antara aktor (pengguna atau elemen luar) dengan sistem. Penggunaan Diagram Use Case bermanfaat dalam merinci kebutuhan sistem, berkomunikasi dengan klien mengenai desain sistem, dan merencanakan uji coba untuk semua fitur system (Dharwiyanti, 2003). Simbol-simbol yang digunakan dalam Diagram Use Case dapat ditemukan dalam Tabel 2.1 berikut ini (Astuti, 2009).

Tabel 2.1 Simbol dalam Use Case Diagram

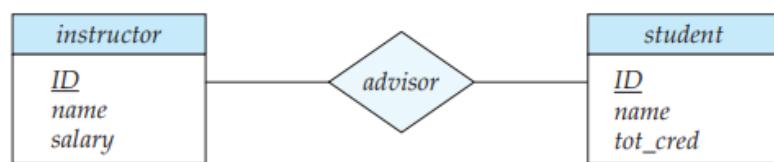
No	Gambar	Nama	Keterangan
1		Use Case	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal di awal frase nama use case
2		Aktor	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri
3		Asosiasi	Komunikasi antara aktor dan use case yang berpartisipasi pada use case atau use case memiliki interaksi dengan aktor
4		Ekstensi	Relasi use case tambahan ke sebuah use case di mana use case yang ditambahkan dapat berdiri sendiri walau tanpa use case tambahan

Tabel 2.1 Simbol dalam *Use Case Diagram* (lanjutan)

5		Generalisasi	Hubungan generalisasi dan spesialisasi (umum - khusus) antara dua buah use case dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya
6		Include	relasi use case tambahan ke sebuah use case di mana use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankan use case ini

### 2.5.3 Entity Relationship Diagram(ERD)

Menurut Pulungan (Pulungan *et al.*, 2023) menyatakan bahwa ERD merupakan salah satu diagram utama representasi model data konseptual yang mencerminkan persyaratan data pengguna dalam sistem basis data.. Menurut ('Afiifah, Azzahra and Anggoro, 2022), ERD ini memrepresentasikan bagaimana entitas saling terkait antara satu dengan yang lainnya dalam database. ERD digunakan untuk pemodelan basis data relasional. ERD dibentuk dari beberapa komponen yang saling berhubungan, dapat dilihat pada gambar 2.9 yang menggambarkan ERD antara instruktur dan pelajar (Silberschatz, 2011)



Gambar 2.9 ERD hubungan instruktur dan pelajar (Silberschatz, 2011)

ERD pada gambar 2.9 diatas terdiri dari beberapa komponen sebagai berikut

1) Entitas :

Entitas adalah "benda" atau "objek" di dunia nyata yang dapat dibedakan dari objek lainnya. Misalnya, setiap orang di universitas adalah entitas.

2) Relasi :

Relasi adalah asosiasi di antara beberapa entitas. Sebagai contoh, kita bisa mendefinisikan relasi "*advisor*" yang mengasosiasikan instruktur A dengan mahasiswa universitas A, yang menyatakan bahwa instruktur A adalah pembimbing mahasiswa A

3) Atribut :

Atribut adalah properti atau karakteristik dari suatu entitas. Setiap atribut memiliki sekumpulan nilai yang disebut domain. Contoh seseorang memiliki atribut nama

## **BAB III**

### **METODE PENELITIAN**

Bab ini menjelaskan tentang objek penelitian, metode pengumpulan data, dan flowchart penelitian pada pengembangan sistem informasi manajemen klinik gigi Xenon Dental House.

#### **3.1 Objek Penelitian**

Klinik gigi Xenon Dental House, yang berlokasi di Jl. Palembang No 11, Ulak Karang, Kota Padang, merupakan pusat pelayanan kesehatan gigi yang pada awalnya didirikan pada tahun 2020 melalui kolaborasi beberapa mahasiswa kedokteran gigi. Klinik ini awalnya berfokus pada penyediaan peralatan dan keperluan perawatan gigi, namun seiring perkembangannya, telah berkembang menjadi penyedia berbagai layanan kesehatan gigi. Proses pelayanan di klinik ini dimulai dengan reservasi oleh pasien melalui kunjungan langsung atau melalui aplikasi WhatsApp milik Klinik gigi Xenon Dental House. Data identitas dan keluhan pasien dicatat, dan pasien diberikan jadwal untuk tindakan dokter. Sehari sebelum jadwal tindakan, admin klinik mengirimkan pengingat kepada pasien. Saat pasien datang, tindakan medis dilakukan sesuai penilaian dokter, dan pasien dijadwalkan untuk kunjungan kontrol jika diperlukan. Jenis tindakan yang disediakan oleh klinik meliputi konsultasi dokter, tambal gigi, pencabutan gigi, scaling gigi, dan pemasangan behel.

Klinik ini awalnya berfokus pada penyediaan peralatan dan keperluan perawatan gigi. Namun, seiring perkembangannya, Klinik gigi Xenon Dental House mulai menyediakan berbagai layanan kesehatan gigi dan telah berkembang pesat dengan kini memiliki cabang di dua kota, yaitu Kota Padang dan Kota Bukittinggi. Kolaborasi antara beberapa dokter gigi membentuk dasar keberhasilan klinik ini sebagai sebuah organisasi. Fokus awal klinik terhadap peralatan dan perawatan gigi telah berkembang seiring waktu, mengakomodasi berbagai layanan kesehatan gigi. Proses pelayanan yang melibatkan reservasi, pencatatan data pasien,

penjadwalan, dan tindakan medis sesuai dengan penilaian. Jenis layanan yang ada seperti konsultasi dokter, tambal gigi, pencabutan gigi, scaling gigi, dan pemasangan behel.

### **3.2 Metode Pengumpulan Data**

Dalam proses pengumpulan data untuk pengembangan aplikasi ini, metode yang digunakan mencakup studi literatur dan studi lapangan. Studi lapangan melibatkan observasi, wawancara, dan analisis dokumen.

#### **3.2.1 Observasi**

Pengumpulan data dimulai dengan observasi, yang memungkinkan peneliti untuk memeriksa secara langsung operasi bisnis yang terjadi di Xenon Dental House. Dalam tahap observasi ini, proses bisnis yang diobservasi meliputi reservasi pasien, pengelolaan data pasien, serta aspek keuangan klinik diamati secara teliti.

#### **3.2.2 Wawancara**

Pendekatan wawancara digunakan untuk menggali pemahaman yang lebih dalam mengenai proses bisnis yang sedang berlangsung di Xenon Dental House. Wawancara dengan pihak terkait di klinik bertujuan untuk mendapatkan informasi yang lebih rinci mengenai penelitian yang dilakukan. Wawancara dilakukan dengan memberikan pertanyaan kepada pemilik dari Xenon Dental House terkait kendala dan kebutuhan pada proses bisnis Xenon Dental House. Wawancara dilakukan untuk mengetahui bagaimana alur proses pengelolaan data pasien, dokter, keuangan, pengelolaan data rekam medis, pengelolaan jadwal rawat pasien, dan pembayaran

#### **3.2.3 Analisis Dokumen**

Melalui analisis dokumen, data diperoleh dengan memeriksa dan mengevaluasi berbagai dokumen seperti dokumen data pasien, pencatatan penjadwalan reservasi pasien serta keuangan yang telah terkumpul dari objek penelitian. Dokumen-dokumen ini berkaitan dengan sistem informasi manajemen klinik gigi di Xenon

Dental House, seperti catatan pasien, laporan keuangan, dan prosedur internal klinik.

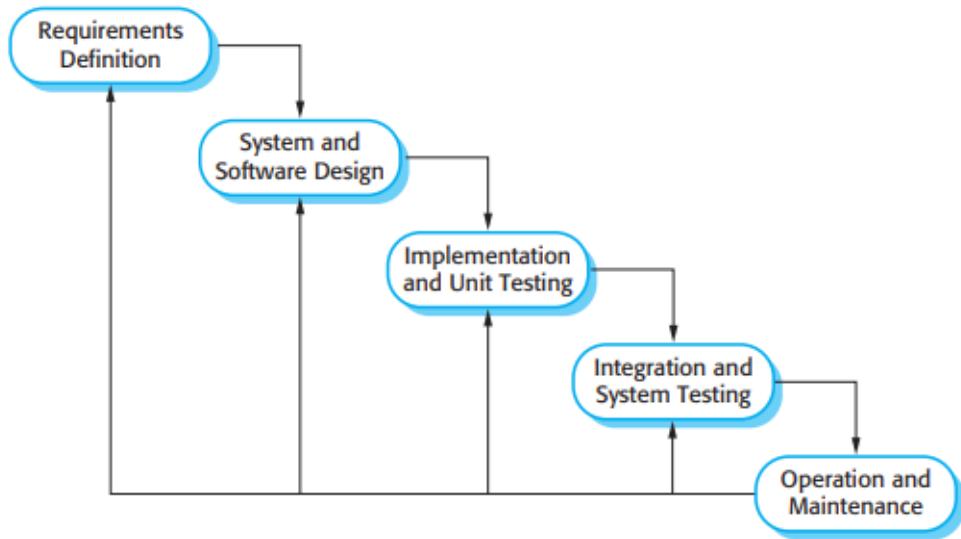
### **3.2.4 Studi Literatur**

Studi literatur mencakup pengumpulan dan telaah berbagai sumber literatur yang relevan dengan penelitian ini. Informasi yang diambil dari literatur diperoleh dari berbagai sumber, jurnal, buku, jurnal ilmiah, penelitian sebelumnya termasuk situs internet yang berkaitan dengan manajemen klinik gigi.

Pendekatan yang komprehensif ini memastikan bahwa data yang diperoleh dalam penelitian ini adalah sesuai dengan kebutuhan analisis dan pengembangan aplikasi, serta berdasarkan pemahaman mendalam tentang praktik bisnis di Xenon Dental House dan pengetahuan yang ada dalam literatur ilmiah.

### **3.3 Metode Pengembangan**

Metode yang digunakan dalam pengembangan aplikasi ini mengikuti pendekatan waterfall. Model pengembangan ini bersifat linear dari tahap awal pengembangan sistem yaitu tahap perencanaan sampai tahap akhir pengembangan sistem yaitu tahap pemeliharaan. Tahapan berikutnya tidak akan dilaksanakan sebelum tahapan sebelumnya selesai dilaksanakan dan tidak bisa kembali atau mengulang ke tahap sebelumnya (Pricillia and Zulfachmi, 2021). Kelebihan yang terkait dengan metode waterfall ini adalah Tahapan proses pengembangannya tetap (pasti), mudah diaplikasikan, dan prosesnya teratur dan cocok digunakan untuk produk software/program yang sudah jelas kebutuhannya. Namun, metode waterfall juga memiliki kelemahan, yaitu Terjadinya pembagian proyek menjadi tahap-tahap yang tidak fleksibel, karena komitmen harus dilakukan pada tahap awal proses (Pricillia and Zulfachmi, 2021). Fase-fase pada pelaksanaan metode waterfall dapat dilihat pada Gambar 3.1 :



Gambar 3.1 Waterfall Model(Sommerville, 2011)

Dalam konteks pengembangan sistem informasi manajemen klinik gigi, proses hanya mencapai tahap keempat, yaitu tahap integrasi dan pengujian sistem. Hal ini disebabkan karena beberapa faktor, termasuk fokus pengerjaan tugas akhir ini adalah perancangan, pengembangan sistem, keterbatasan waktu yang ketat dan sumber daya yang terbatas jika dibandingkan dengan proyek komersial. Penjelasan untuk setiap tahap adalah sebagai berikut (Sommerville, 2011):

1. Analisis Kebutuhan dan Pendefinisian (Requirement Analysis and Definition)

Pada tahap ini, dilakukan analisis dan pengumpulan data untuk mengidentifikasi kebutuhan sistem berdasarkan data yang telah dikumpulkan. Pendekatan ini melibatkan analisis dokumen, wawancara, observasi, dan studi literatur yang relevan dengan pengembangan sistem informasi manajemen klinik gigi di Xenon Dental House. Hasil yang didapatkan dari tahap ini adalah BPMN, daftar fungsional pengguna, *use case diagram* dan *sequence diagram*.

2. Perancangan Sistem dan Perangkat Lunak (System and Software Design)

Tahap ini berkaitan dengan perencanaan solusi perangkat lunak. Data yang diperoleh selama tahap analisis digunakan untuk merancang sistem baru dengan mempertimbangkan struktur data, arsitektur perangkat lunak, perancangan basis

data, dan antarmuka aplikasi. Hasil yang didapatkan dari tahap ini adalah rancangan basis data, arsitektur aplikasi, class diagram dan mockup aplikasi.

### 3. Implementasi dan Pengujian Unit (Implementation and Unit Testing)

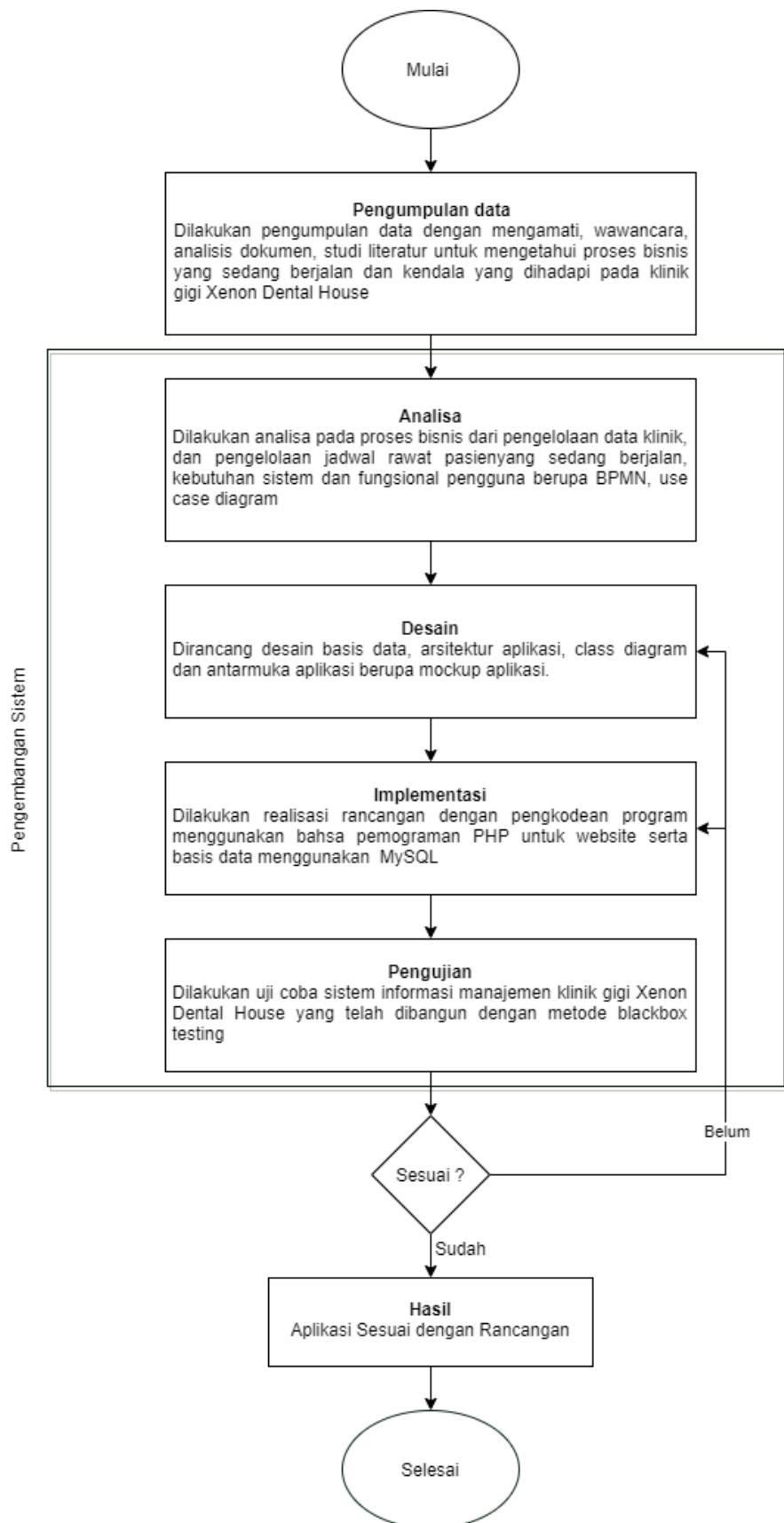
Tahapan ini adalah tahapan pembuatan program dan database dari design program dan design database yang sudah dibuat di tahap sebelumnya. Setiap modul program yang sudah dibuat akan diuji untuk menguji secara fungsionalitasnya. Dari hasil perancangan sistem, *requirement* perangkat lunak diubah menjadi kode pemrograman dalam aplikasi web menggunakan bahasa pemograman PHP framework Laravel serta menerapkan pendekatan Progressive Web Apps(PWA). Basis data yang digunakan dalam pembuatan web ini adalah MySQL. Setelah penulisan kode selesai, langkah berikutnya adalah tahap pengujian unit. Pengujian unit melibatkan verifikasi bahwa setiap unit memenuhi spesifikasinya.

### 4. Integrasi dan Pengujian sistem ( Integration and System testing)

Tahap ini adalah tahapan unit-unit program atau program-program individual diintegrasikan dan diuji sebagai suatu sistem yang lengkap untuk memastikan bahwa persyaratan perangkat lunak telah terpenuhi. Pengujian sistem harus difokuskan pada pengujian interaksi komponen.

#### 3.4 Flowchart Penelitian

Berdasarkan metode pengembangan sistem disusun sebuah flowchart atau diagram alur proses penelitian yang menggambarkan tahapan dalam pembangunan aplikasi manajemen laporan kriminal. Tahap-tahap dalam penelitian ini diilustrasikan dalam gambar 3.2 dibawah ini.



Gambar 3.2 Tahapan Penelitian

## **BAB IV**

### **ANALISIS DAN PERANCANGAN**

Bab ini membahas hasil analisis yang dilakukan terhadap kebutuhan perancangan sistem informasi manajemen klinik gigi berbasis web di klinik gigi Xenon Dental House. Analisis tersebut melibatkan penggunaan BPMN(Business Process Model and Notation), daftar fungsional pengguna, use case diagram dan sequence diagram. Sementara dalam fase perancangan sistem, digunakan entity relationship diagram (ERD), struktur database, dan antarmuka pengguna.

#### **4.1 Analisis Sistem**

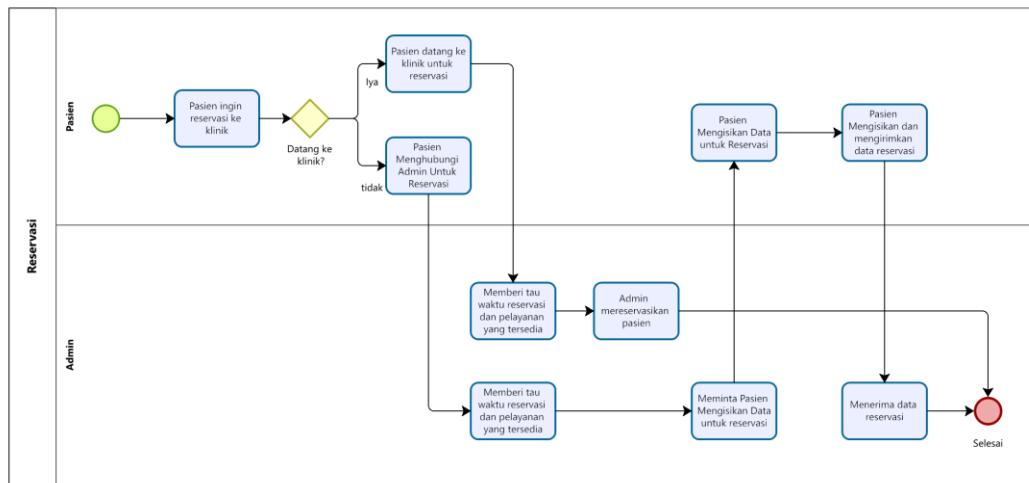
Analisis sistem merupakan proses memeriksa sistem yang sedang berjalan dan merancang sistem yang diusulkan agar memenuhi kebutuhan fungsional yang diinginkan dari sistem yang akan dibangun. Analisis sistem ini disusun dengan menggunakan BPMN, dan UML (Unified Modeling Language). UML yang diterapkan dalam analisis sistem ini meliputi sequence diagram dan use case diagram.

##### **4.1.1 Analisis sistem yang Sedang Berjalan**

Sistem yang sedang beroperasi ini diperoleh dari hasil pengumpulan data dengan cara analisis dokumen, wawancara, observasi dan analisis yang telah dilakukan di klinik gigi Xenon Dental House. Penggambaran atau pemodelan sistem yang berjalan akan disajikan menggunakan Business Process Model Notation terdiri dari system reservasi dan pelayanan, sistem manajemen jadwal.

###### **4.1.1.1 Sistem Reservasi**

Sistem reservasi pada klinik gigi xenon dental house dapat dilihat dalam Gambar 4.1.



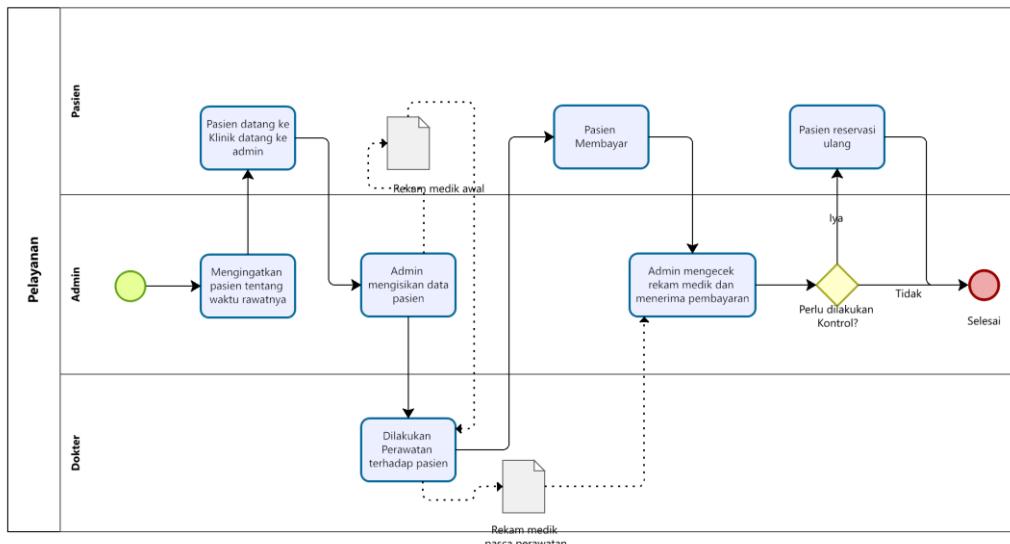
Gambar 4.1 BPMN Sistem reservasi

Berikut ini adalah proses bisnis reservasi yang sedang berjalan pada klinik gigi Xenon Dental House berdasarkan gambar 4.1.

1. Pasien bisa reservasi dengan cara datang reservasi langsung ke klinik gigi atau reservasi dengan menghubungi admin melalui aplikasi WhatsApp
2. Jika melalui whatsapp admin memberitahu waktu dan pelayanan yang tersedia serta meminta pasien untuk mengisi data untuk reservasi
3. Setelah pasien mengisikan data reservasi dan memberikannya kepada admin, pasien akan menunggu sesuai waktu yang sudah direservasi.
4. Jika pasien datang ke klinik, maka admin akan langsung membantu mereservasi jadwal pasien
5. Proses selesai

#### 4.1.1.2 Sistem Pelayanan

Sistem pelayanan pada klinik gigi xenon dental house dapat dilihat dalam Gambar 4.2.



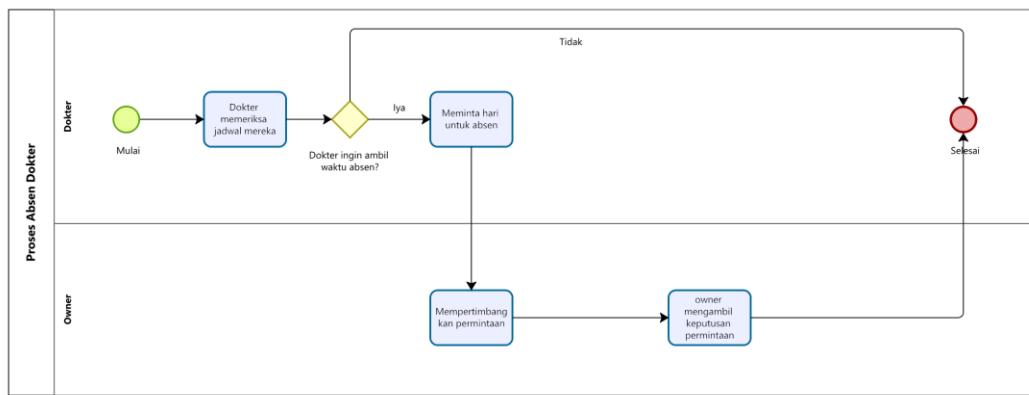
Gambar 4.2 BPMN Sistem pelayanan

Berikut ini adalah proses bisnis pelayanan yang sedang berjalan pada klinik gigi Xenon Dental House berdasarkan gambar 4.2.

1. Sebelum pasien dirawat, admin akan mengingatkan pasien untuk datang ke klinik untuk dilakukan perawatan yang sudah direservasi.
2. Pasien datang ke klinik dan diminta melakukan pendataan Kesehatan sebelum dilakukannya perawatan
3. Dokter melakukan perawatan terhadap pasien sesuai dengan kondisi pasien serta menuliskan kondisi pasien pada rekam medis.
4. Setelah perawatan selesai, pasien melakukan pembayaran
5. Jika dibutuhkan kontrol setelah perawatan maka pasien akan melakukan reservasi ulang kepada admin
6. Jika tidak diperlukan kontrol maka bisnis proses selesai

#### 4.1.1.3 Sistem perizinan dokter

Sistem perizinan dokter pada klinik gigi Xenon Dental House saat ini dapat dilihat dalam Gambar 4.3.



Gambar 4.3 BPMN manajemen jadwal

Berikut ini adalah proses bisnis perizinan dokter yang sedang berjalan pada klinik gigi Xenon Dental House berdasarkan gambar 4.3.

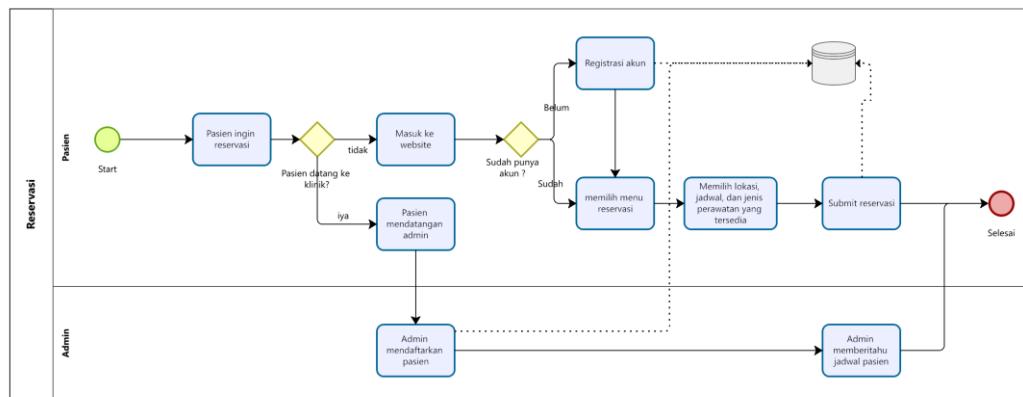
1. Dokter memeriksa jadwal mereka
2. Jika ingin mengambil absen, dosen akan meminta izin untuk absen kepada owner dari klinik di hari tertentu
3. Jika diizinkan maka dokter dibolehkan libur pada hari tersebut
4. Jika tidak diizinkan maka dokter tidak mendapatkan libur
5. Proses bisnis selesai

#### 4.1.2 Sistem yang Diusulkan

Dari sistem yang sudah berjalan, diusulkan sistem baru dengan membangun sistem informasi manajemen klinik gigi yang berbasis kan *website* pada klinik gigi Xenon Dental House. Sistem ini dimodelkan dengan BPMN, terdiri dari sistem reservasi dan pelayanan, sistem perizinan dokter.

##### 4.1.2.1 Sistem Reservasi

System reservasi dan pelayanan yang diusulkan dapat dilihat seperti pada gambar 4.4



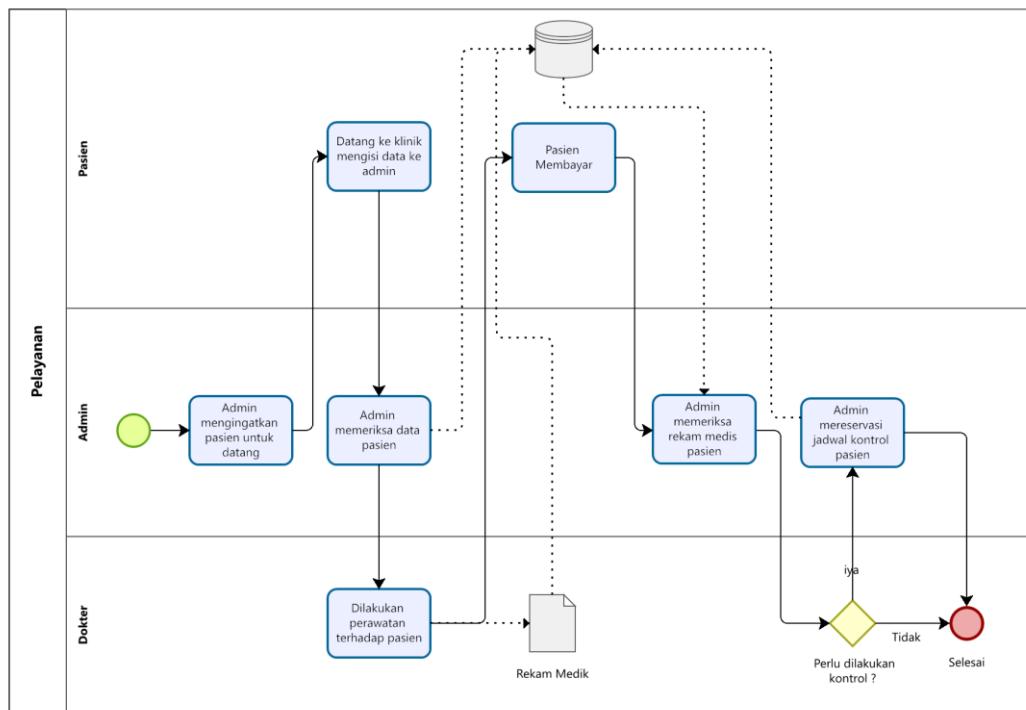
Gambar 4.4 BPMN Sistem reservasi

Berikut ini adalah proses bisnis reservasi dan pelayanan yang diusulkan pada klinik gigi Xenon Dental House berdasarkan gambar 4.4.

1. Pasien bisa reservasi dengan cara datang reservasi langsung ke klinik gigi atau reservasi dengan menggunakan *website* klinik gigi Xenon Dental House
2. Jika pasien datang ke klinik, maka pasien akan melakukan reservasi pada admin sesuai dengan jadwal yang tersedia melalui akun admin
3. Jika melalui *website*, maka pasien perlu membuka *website* klinik gigi untuk melakukan reservasi terlebih dahulu
4. Jika sudah memiliki akun pada *website* maka pasien bisa login di *website* klinik, jika belum registrasi akun terlebih dahulu dan melakukan login setelahnya
5. Pasien melakukan reservasi dengan memilih lokasi, jadwal dan jenis perawatan yang tersedia pada klinik gigi
6. Jika sudah, pasien mensubmit reservasi
7. Proses selesai

#### 4.1.2.2 Sistem Pelayanan

System pelayanan yang diusulkan pada klinik xenon dental house dapat dilihat seperti pada gambar 4.5



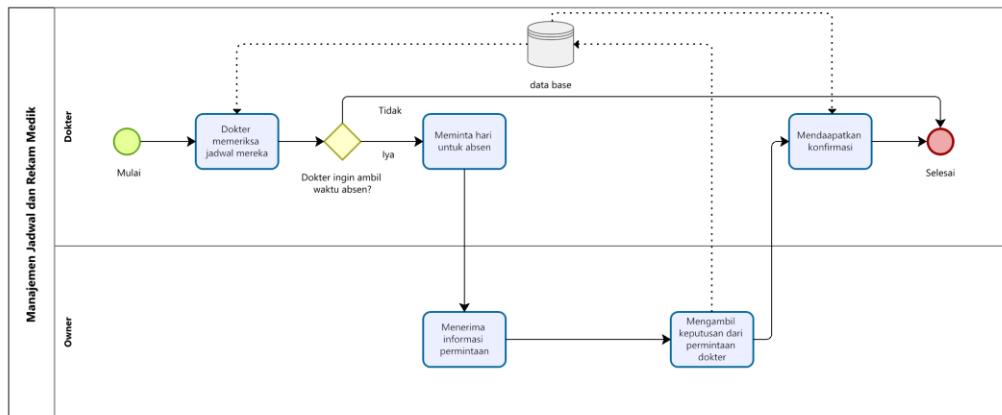
Gambar 4.5 BPMN Sistem pelayanan yang diusulkan

Berikut ini adalah proses bisnis reservasi dan pelayanan yang diusulkan pada klinik gigi Xenon Dental House berdasarkan gambar 4.3.

1. Sebelum pasien dirawat, admin akan mengingatkan pasien untuk datang ke klinik untuk dilakukan perawatan yang sudah direservasi.
2. Pasien datang ke klinik dan admin memeriksa data pasien
3. Dokter melakukan perawatan yang diperlukan terhadap pasien dan dokter menambahkan data rekam medis sesuai dengan kondisi pasien
4. Setelah selesai, pasien membayar
5. Admin memeriksa rekam medik pasien dan menerima pembayaran pasien
6. Jika diperlukan, admin mereservasi ulang untuk jadwal pasien
7. Jika tidak diperlukan maka proses bisnis selesai

#### 4.1.2.3 Sistem perizinan dokter

Sistem perizinan dokter pada klinik gigi Xenon Dental House yang diusulkan dapat dilihat dalam Gambar 4.6



Gambar 4.6 BPMN Perizinan dokter

Berikut ini adalah proses bisnis perizinan dokter yang diusulkan pada klinik gigi Xenon Dental House berdasarkan gambar 4.6.

1. Dokter login ke *website*
2. Dokter memeriksa jadwal mereka pada *website*
3. Jika ingin mengambil absen, dosen akan meminta izin untuk absen kepada owner dari klinik di hari tertentu
4. Jika diizinkan maka dokter dibolehkan libur dan dokter menginputkan data tersebut pada *website*
5. Jika tidak diizinkan maka dokter tidak mendapatkan libur
6. Proses bisnis selesai

#### 4.1.3 Analisis Kebutuhan Fungsional

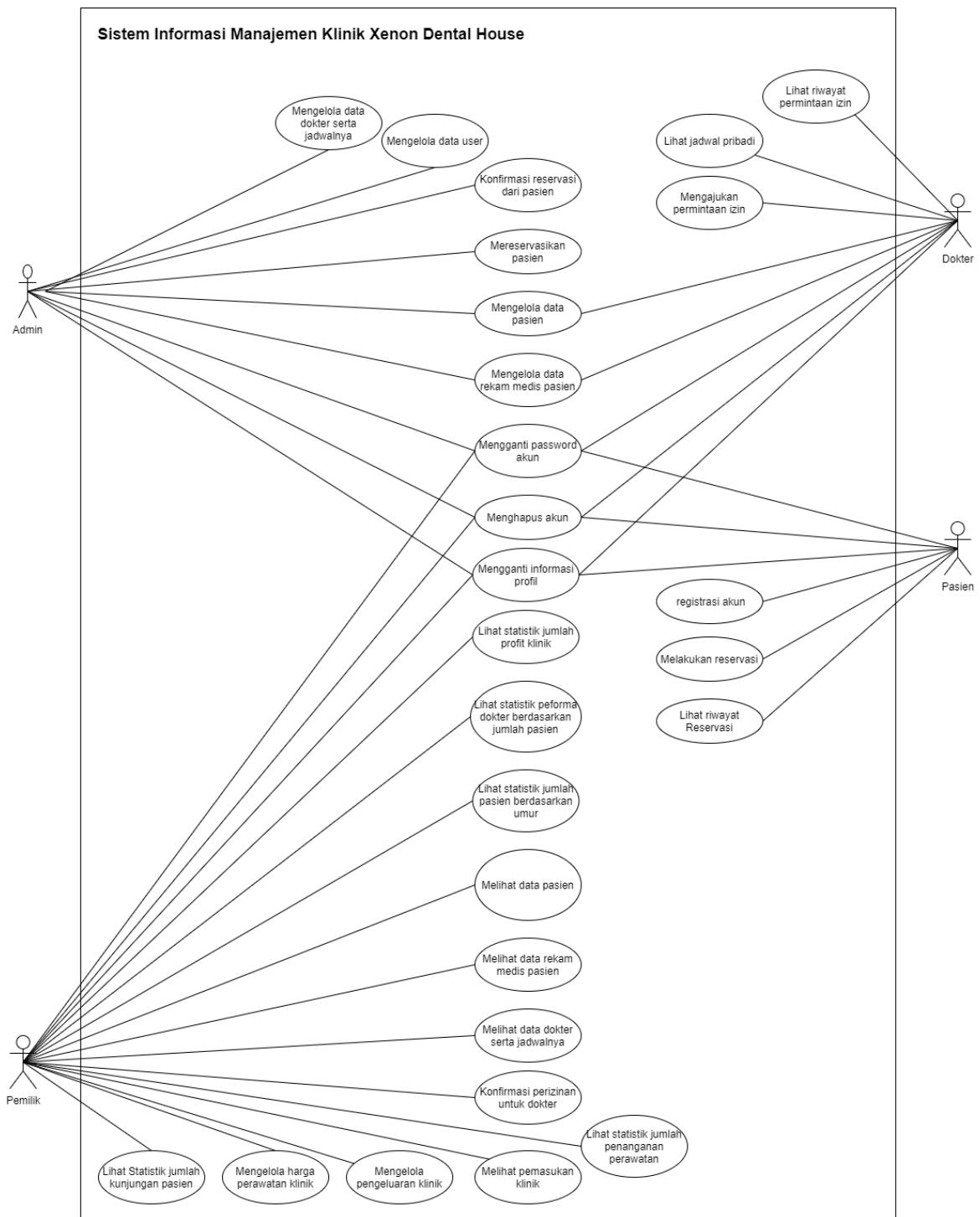
Dari analisis alur proses *website* sistem informasi manajemen klinik gigi yang diajukan, beberapa kebutuhan fungsional dapat dirumuskan, antara lain:

1. User dapat login pada *website*
2. User dapat mengganti password akun mereka
3. User dapat mengganti informasi profil mereka
4. User dapat menghapus akun mereka
5. Admin dapat melihat dashboard
6. Admin dapat mengkonfirmasi reservasi dari pasien
7. Admin dapat membantu reservasi pasien di klinik
8. Admin dapat mengelola data pasien
9. Admin dapat mengelola data rekam medis pasien
10. Admin dapat mengelola data dokter serta jadwal dokter
11. Admin dapat mengelola data data user
12. Pasien dapat registrasi akun
13. Pasien dapat melakukan reservasi perawatan
14. Pasien dapat melihat riwayat reservasi
15. Dokter dapat melihat jadwal mereka
16. Dokter dapat mengelola data pasien
17. Dokter membuat permintaan izin pada hari tertentu
18. Dokter dapat melihat permintaan izin mereka
19. Pemilik dapat melihat jumlah kunjungan pasien
20. Pemilik dapat melihat jumlah profit klinik
21. Pemilik dapat melihat jumlah penanganan perawatan
22. Pemilik dapat melihat performa dokter berdasarkan jumlah pasien
23. Pemilik bisa melihat data pasien berdasarkan umurnya

24. Pemilik bisa melihat data dokter
25. Pemilik bisa melihat data pasien
26. Pemilik bisa melihat data pemasukan
27. Pemilik bisa memutuskan perizinan dokter
28. Pemilik dapat mengelola pencatatan pengeluaran klinik
29. Pemilik dapat mengelola harga perawatan

#### **4.1.4 Use Case Diagram**

Berdasarkan analisis kebutuhan fungsional yang telah dilakukan pada sub bab sebelumnya, setiap kebutuhan fungsional tersebut dihubungkan dengan aktor yang terlibat dalam sistem, yang kemudian dimodelkan menjadi use case diagram. Setiap aktor memiliki hak akses terhadap fungsionalitas sistem yang berbeda sesuai dengan peran dan tanggung jawabnya masing-masing. Semua fungsionalitas ini bisa diakses setelah user melakukan login. Use case diagram ini dapat dilihat pada Gambar 4.7 di bawah ini.



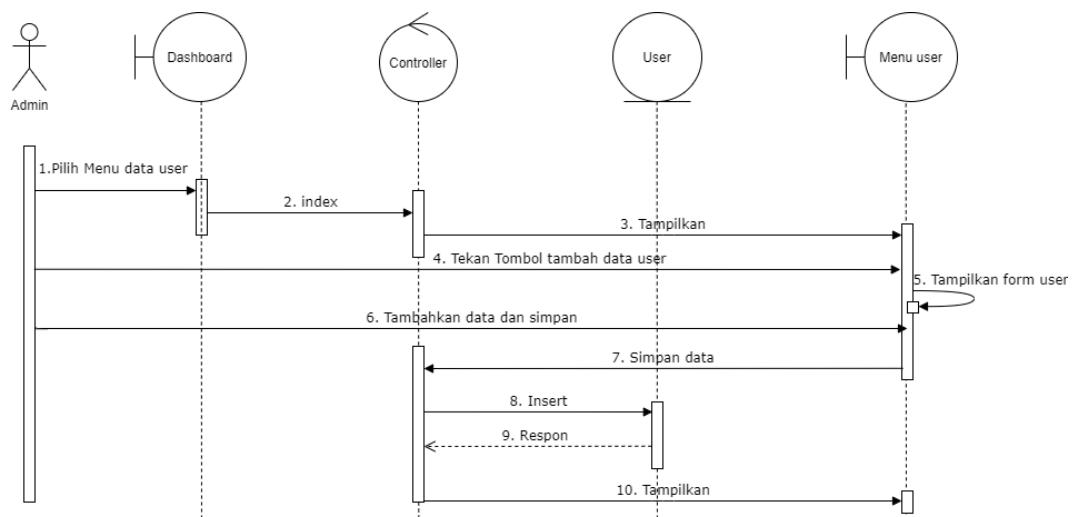
Gambar 4.7 Use Case Diagram Sistem

#### 4.1.5 Sequence Diagram

Sequence diagram menggambarkan aliran pesan dan data dalam proses interaksi antar objek yang terlibat dalam sistem secara berurutan. Sequence diagram yang akan dibahas di subbab ini mencakup proses login user, reservasi, proses menambahkan data user, dashboard owner . Sequence diagram lainnya dapat ditemukan pada Lampiran B.

##### 5.2.2.4 Sequence Diagram Menginputkan Data User

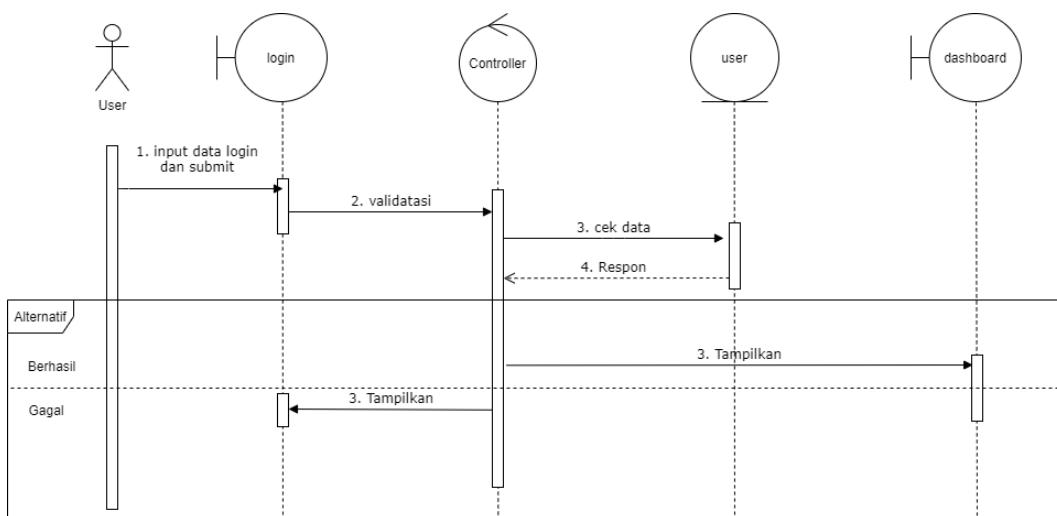
Sequence diagram ini menampilkan proses interaksi antara view, control, entity class saat dilakukannya penginputan data baru user pada sistem. Proses ini dimulai ketika user memilih menu data user pada tampilan dashboard. Setelah itu controller akan mengarahkan user ke tampilan menu user. Setelah berada pada view menu user, user akan menekan tombol tambah data user dan view akan menampilkan form user yang mana pada view ini user bisa menginputkan data. Pada tampilan form ini, user akan mengisi form dengan data user baru yang akan ditambahkan. Setelah itu controller request untuk dilakukannya penambahan data baru pada database. Setelah selesai maka data yang baru ditambahkan akan muncul pada tampilan menu user.



Gambar 4.8 Sequence Diagram input data user

#### 4.1.5.2 Sequence Diagram Login

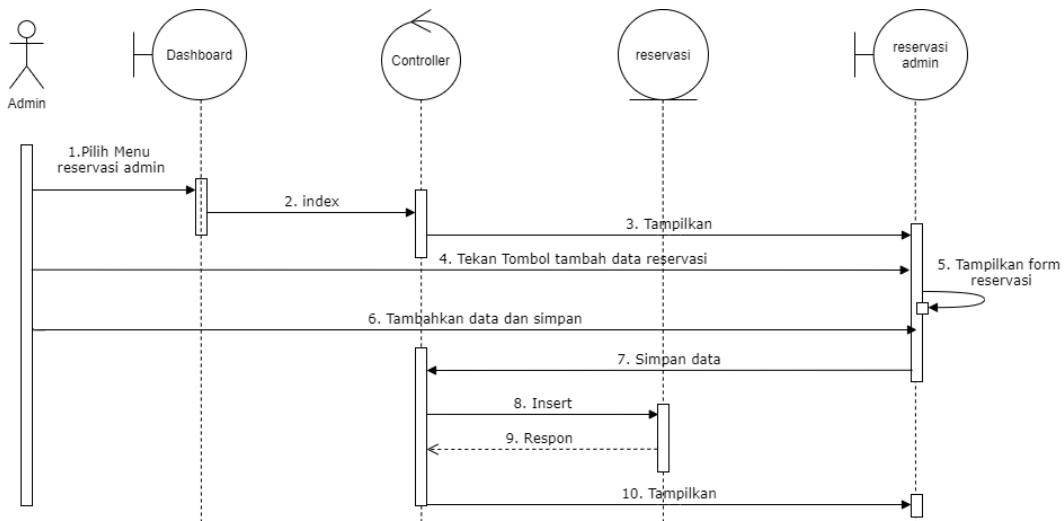
Sequence diagram ini menampilkan proses antara view, control, entity class saat dilakukannya proses login pada system ini. Proses ini dimulai dengan user menginputkan email dan passwordnya pada form login. Setelah selesai, user menekan tombol login dan hasil dari pengisian tersebut akan dilakukan validasi. Sistem akan memberikan respon, jika berhasil maka sistem akan mengarahkan user pada tampilan dashboard dan jika gagal maka sistem akan menampilkan gagal dari pengisian data login.



Gambar 4.9 Sequence diagram login

#### 4.1.5.3 Sequence Diagram Reservasi Admin

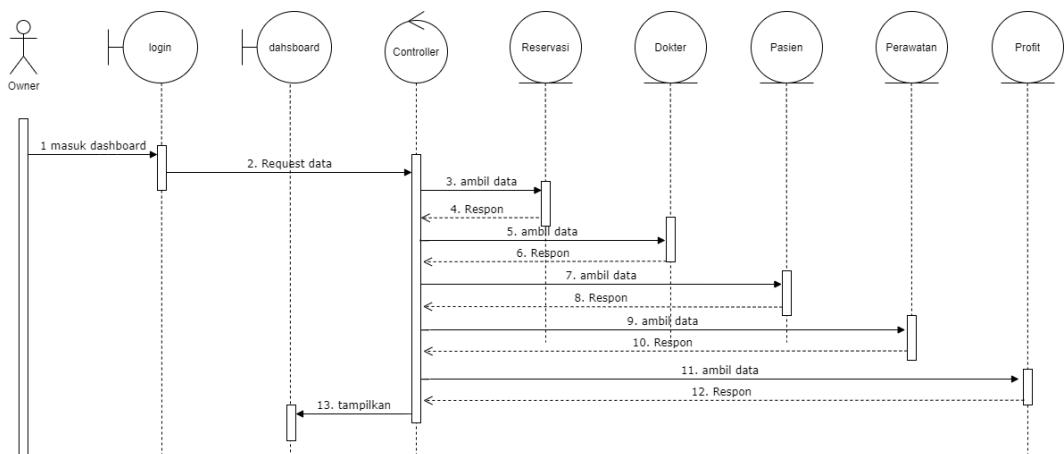
Sequence diagram ini menampilkan proses antara view, control, entity class saat dilakukannya proses reservasi pasien oleh user admin. Proses dimulai dengan admin memilih menu reservasi admin pada tampilan dashboard. Setelah itu controller akan mengarahkan admin ke tampilan reservasi admin. Pada tampilan admin user akan menekan tombol tambah data reservasi dan sistem akan menampilkan form reservasi untuk user menginputkan reservasi baru. Admin mengisi semua data yang diperlukan pada form dan setelah itu user menekan tombol simpan. Controller akan melakukan request untuk dilakukannya penambahan data baru pada database. Terakhir sistem akan menampilkan list data reservasi yang mana telah ditambahkannya data baru.



Gambar 4.10 Sequence diagram reservasi admin

#### 4.1.5.4 Sequence Diagram Menampilkan Dashboard Owner

Sequence diagram ini menampilkan proses antara view, control, entity class saat dilakukannya proses menampilkan dashboard owner. Proses awalnya adalah owner pergi ke halaman login dan melakukan login. Setelah login maka controller akan melakukan request untuk data data yang diperlukan karena pada dashboard terdapat berbagai analisis data. Data yang diambil berupa data reservasi, dokter, pasien, perawatan, dan profit. Setelah data di request maka owner akan diarahkan ke dashboard dana data yang telah diambil akan divisualisasikan pada tampilan dashboard.



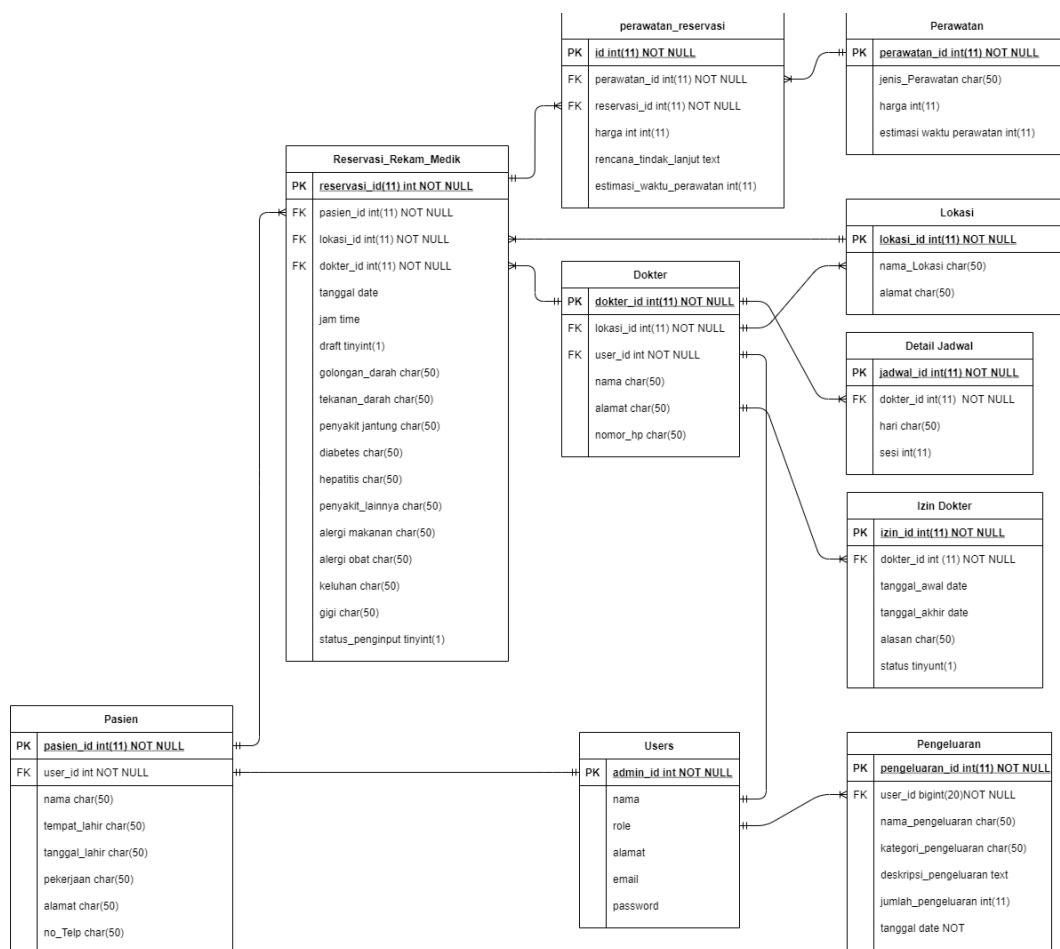
Gambar 4.11 Sequence diagram lihat dashboard owner

## 4.2 Perancangan sistem

Dari tahapan analisis proses yang sedang berjalan, serta alur sistem dan kebutuhan fungsional sistem yang akan dibangun, diperoleh hasil yang menjadi landasan dan tolok ukur untuk merancang sistem. Proses perancangan sistem ini mencakup rancangan basis data, arsitektur aplikasi, class diagram dan perancangan antarmuka.

### 4.2.1 Perancangan Basis data

Struktur tabel menggambarkan setiap tabel yang telah dirancang dalam ERD (Entity Relationship Diagram) dengan menjelaskan semua atribut yang ada di masing-masing tabel. Atribut yang akan diuraikan untuk setiap tabel mencakup nama tabel, primary key, foreign key, nama atribut, tipe data. Berikut ini adalah rincian struktur tabel yang terdapat dalam ERD tersebut.



Gambar 4.12 ERD Sistem Informasi Manajemen Klinik

#### **5.2.2.4 Struktur Tabel Perawatan**

Tabel perawatan adalah tabel yang berisikan detail perawatan yang ada pada klinik gigi Xenon Dental House. Pada tabel ini terdapat primary key dengan nama perawatan\_id. Struktur tabel ini dapat dilihat pada tabel 4.1 berikut

Tabel 4.1 Tabel Perawatan

Nama Atribut	Tipe Data	Ukuran	Keterangan
Perawatan_id	int	11	Primary Key
Jenis_perawatan	varchar	50	
Harga	int	11	
estimasi	int	11	

#### **4.2.1.2 Struktur Tabel Lokasi**

Tabel lokasi adalah tabel yang berisikan data detail terkait lokasi klinik beserta cabang yang ada pada klinik gigi Xenon Dental House. Tabel ini memiliki primary key dengan nama atribut lokasi\_id. Struktur tabel ini dapat dilihat pada tabel 4.2 berikut.

Tabel 4.2 Tabel Lokasi

Nama Atribut	Tipe Data	Ukuran	Keterangan
Lokasi_id	int	11	Primary Key
Nama_lokasi	varchar	50	
Alamat	char	50	

#### **4.2.1.3 Struktur Tabel Detail jadwal**

Tabel detail jadwal adalah tabel yang berisikan detail jadwal dokter yang ada pada klinik gigi Xenon Dental House. Tabel ini memiliki primary key dengan nama atribut jadwal\_id dan foreign key dokter\_id. Struktur tabel ini dapat dilihat pada tabel 4.3 berikut.

Tabel 4.3 Tabel Detail jadwal

Nama Atribut	Tipe Data	Ukuran	Keterangan
Jadwal_id	int	11	Primary Key
Dokter_id	int	11	Foreign Key
hari	varchar	20	
ses	int	11	

#### 4.2.1.4 Struktur Tabel Pengeluaran

Tabel pengeluaran adalah tabel yang berisikan data detail terkait pengeluaran klinik. Tabel ini memiliki primary key dengan nama atribut pengeluaran\_id dan foreign key dengan nama atribut user\_id. Struktur tabel ini dapat dilihat pada tabel 4.4 berikut.

Tabel 4.4 Tabel Pengeluaran

Nama Atribut	Tipe Data	Ukuran	Keterangan
Pengeluaran_id	int	11	Primary Key
User_id	bigint	20	Foreign Key
Nama_pengeluaran	varchar	20	
Kategori_pengeluaran	varchar	20	
Deskripsi_pengeluaran	text		
Jumlah_pengeluaran	int	11	
Tanggal	date		

#### 4.2.1.5 Struktur Tabel Izin Dokter

Tabel izin dokter adalah data yang berisikan data terkait izin dokter dalam bekerja. Tabel izin ini memiliki primary key dengan nama atribut izin\_id dan foreign key user\_id dan lokasi\_id. Struktur tabel 4.5 ini dapat dilihat pada tabel berikut.

Tabel 4.5 Tabel Izin Dokter

Nama Atribut	Tipe Data	Ukuran	Keterangan
Izin_id	int	11	Primary key
Dokter_id	int	11	Foreign Key
Tanggal_awal	date		
Tanggal_akhir	date		
Alasan	varchar	50	
Status	tinyint	1	

#### 4.2.1.6 Struktur Tabel Users

Tabel users adalah tabel yang berisikan detail data detail user. Pada tabel user terdapat primary key dengan nama atribut id. Struktur tabel 4.6 ini dapat dilihat pada tabel berikut.

Tabel 4.6 Tabel Users

Nama Atribut	Tipe Data	Ukuran	Keterangan
id	bigint	20	Primary Key
Nama	varchar	50	
Role	varchar	20	
Email	varchar	255	
Password	varchar	255	
Created_at	Timestamp		
Updated_at	Timestamp		

#### 4.2.1.7 Struktur Tabel Dokter

Tabel dokter adalah tabel yang berisikan data detail dokter. Pada tabel ini terdapat primary key dengan nama atribut dokter\_id dan foreign key dengan nama lokasi\_id. Struktur tabel ini dapat dilihat pada tabel 4.7 berikut.

Tabel 4.7 Tabel Dokter

Nama Atribut	Tipe Data	Ukuran	Keterangan
Dokter_id	int	11	Primary Key
Lokasi_id	int	11	Foreign Key
Nama	varchar	50	
Alamat	varchar	50	
Nomor_hp	varchar	20	

#### 4.2.1.8 Struktur Tabel Perawatan\_reservasi

Tabel perawatan\_reservasi adalah tabel yang menghubungkan antara tabel perawatan dan tabel reservasi. Pada tabel ini terdapat primary key dengan nama atribut id dan foreign key dengan nama harga dan estimasi waktu perawatan. Struktur dari tabel ini dapat dilihat pada tabel 4.8 berikut.

Tabel 4.8 Tabel Perawatan\_reservasi

Nama Atribut	Tipe Data	Ukuran	Keterangan
id	int	11	Primary Key
Perawatan_id	int	11	Foreign Key
Reservasi_id	Int	11	Foreign Key
Harga	int	11	
Estimasi_waktu_perawatan	int	11	
Rencana_tindak_lanjut	text		

#### 4.2.1.9 Struktur Tabel Reservasi\_rekam\_medik

Tabel reservasi\_rekam\_medik adalah tabel yang berisikan reservasi dan rekam medik pasien. Pada tabel ini terdapat primary key dengan nama reservasi\_id dan foreign key dengan nama pasien\_id, lokasi\_id, dan dokter\_id. Struktur tabel ini dapat dilihat pada tabel 4.9 berikut.

Tabel 4.9 Tabel Reservasi Rekam Medik

Nama Atribut	Tipe Data	Ukuran	Keterangan
Reservasi_id	int	11	Primary Key
Pasien_id	int	11	Foreign Key
Lokasi_id	int	11	Foreign Key
Dokter_id	int	11	Foreign Key
Tanggal	date		
Jam	time		
draft	tinyint	1	
Golongan_darah	varchar	20	
Tekanan_darah	varchar	20	
Penyakit_jantung	varchar	20	
Diabetes	varchar	20	
Hepatitis	varchar	20	
Penyakit_lainnya	varchar	20	
Alergi_obat	varchar	20	
Alergi_makanan	varchar	20	
Keluahan	varchar	100	
Gigi	varchar	20	
Status_penginput	tinyint	1	

#### 4.2.1.10 Struktur Tabel Pasien

Tabel pasien adalah tabel yang berisikan data detail terkait pasien. Pada tabel ini terdapat primary key dengan nama pasien\_id dan foreign key dengan nama user\_id. Struktur tabel ini dapat dilihat pada tabel 4.10 berikut.

Tabel 4.10 Tabel Pasien

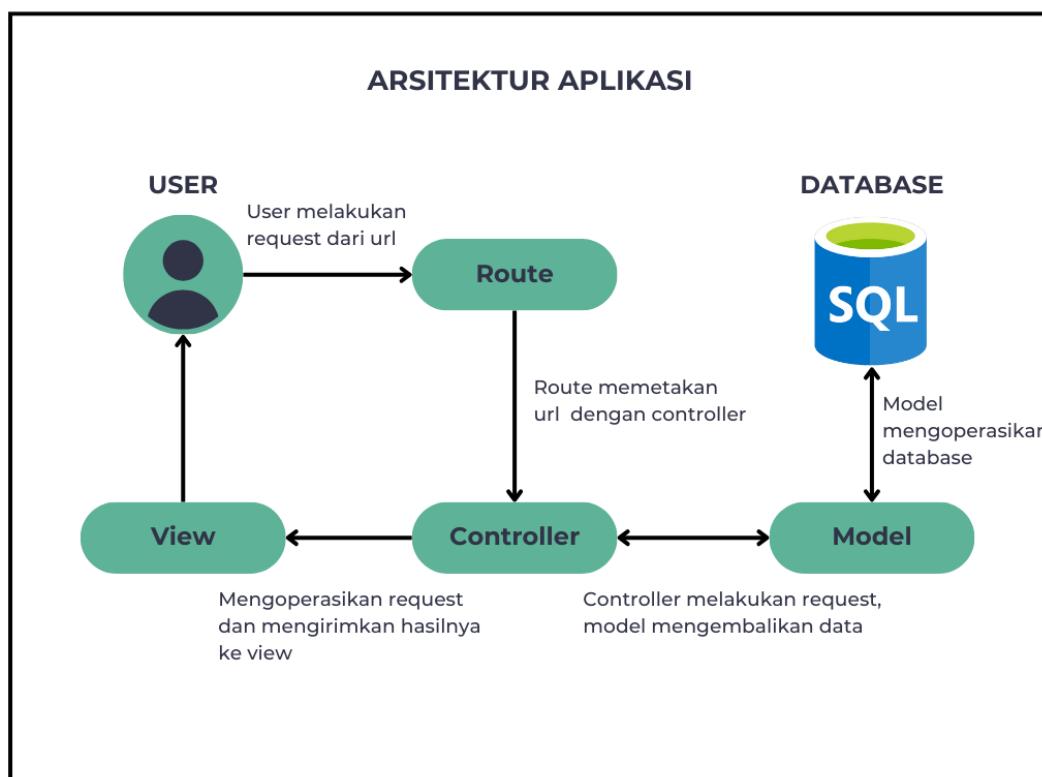
Nama Atribut	Tipe Data	Ukuran	Keterangan
Pasien_id	Int	11	Primary Key
User_id	bigint	20	Foreign Key

Tabel 4.10 Tabel Pasien(lanjutan)

Nama	varchar	50	
Tempat_lahir	varchar	50	
Tanggal_lahir	varchar	20	
Pekerjaan	varchar	50	
Alamat	varchar	20	
No_telp	varchar	20	

#### 4.2.2 Arsitektur Aplikasi

Arsitektur aplikasi yang digunakan dalam penelitian ini mengikuti pola Model-View-Controller (MVC). Pola ini dipilih karena memisahkan logika aplikasi, tampilan, dan manipulasi data, sehingga mempermudah pengembangan dan pemeliharaan aplikasi. Aplikasi ini dikembangkan menggunakan framework Laravel 11 dan menggunakan MySQL sebagai basis data. Gambar pada arsitektur ini bisa dilihat pada gambar 4.13 berikut.



Gambar 4.13 Arsitektur Aplikasi Website

Berikut adalah penjelasan dari setiap komponen dalam arsitektur MVC ini:

1. User (Pengguna):

Pengguna melakukan request dari URL. Ini adalah titik awal di mana pengguna berinteraksi dengan aplikasi web. Setiap permintaan dari pengguna akan diproses oleh sistem melalui serangkaian yang terstruktur.

2. Route:

Route adalah komponen yang bertugas memetakan URL permintaan pengguna ke controller yang sesuai. Dalam Laravel, routing ini didefinisikan dalam file web.php atau api.php, tergantung pada jenis aplikasi yang dikembangkan. Route memastikan bahwa setiap permintaan dari pengguna diteruskan ke controller yang tepat untuk diproses lebih lanjut.

3. Controller:

Controller bertanggung jawab untuk menerima permintaan dari Route, memprosesnya, dan menentukan apa yang harus dilakukan selanjutnya. Controller melakukan request ke Model untuk mengambil atau memanipulasi data dari database. Setelah mendapatkan data yang diperlukan, Controller akan meneruskannya ke View untuk ditampilkan kepada pengguna.

4. Model:

Model adalah komponen yang berinteraksi langsung dengan database. Model bertugas untuk mengoperasikan database, termasuk mengambil, menyimpan, memperbarui, dan menghapus data.

5. View:

View adalah komponen yang bertugas untuk menampilkan data kepada pengguna. View menerima data dari Controller dan menyajikannya dalam bentuk yang dapat dipahami oleh pengguna. Dalam Laravel, Viewnya menggunakan Blade template engine untuk menampilkan halaman HTML.

## Alur Kerja

1. Pengguna mengirimkan request melalui URL.
2. Route memetakan URL tersebut ke Controller yang sesuai.
3. Controller menerima permintaan dari Route dan meminta data dari Model.
4. Model berinteraksi dengan database MySQL untuk mengambil atau memanipulasi data.
5. Model mengembalikan data ke Controller.
6. Controller mengoperasikan data dan mengirimkannya ke View.
7. View menampilkan data kepada pengguna.

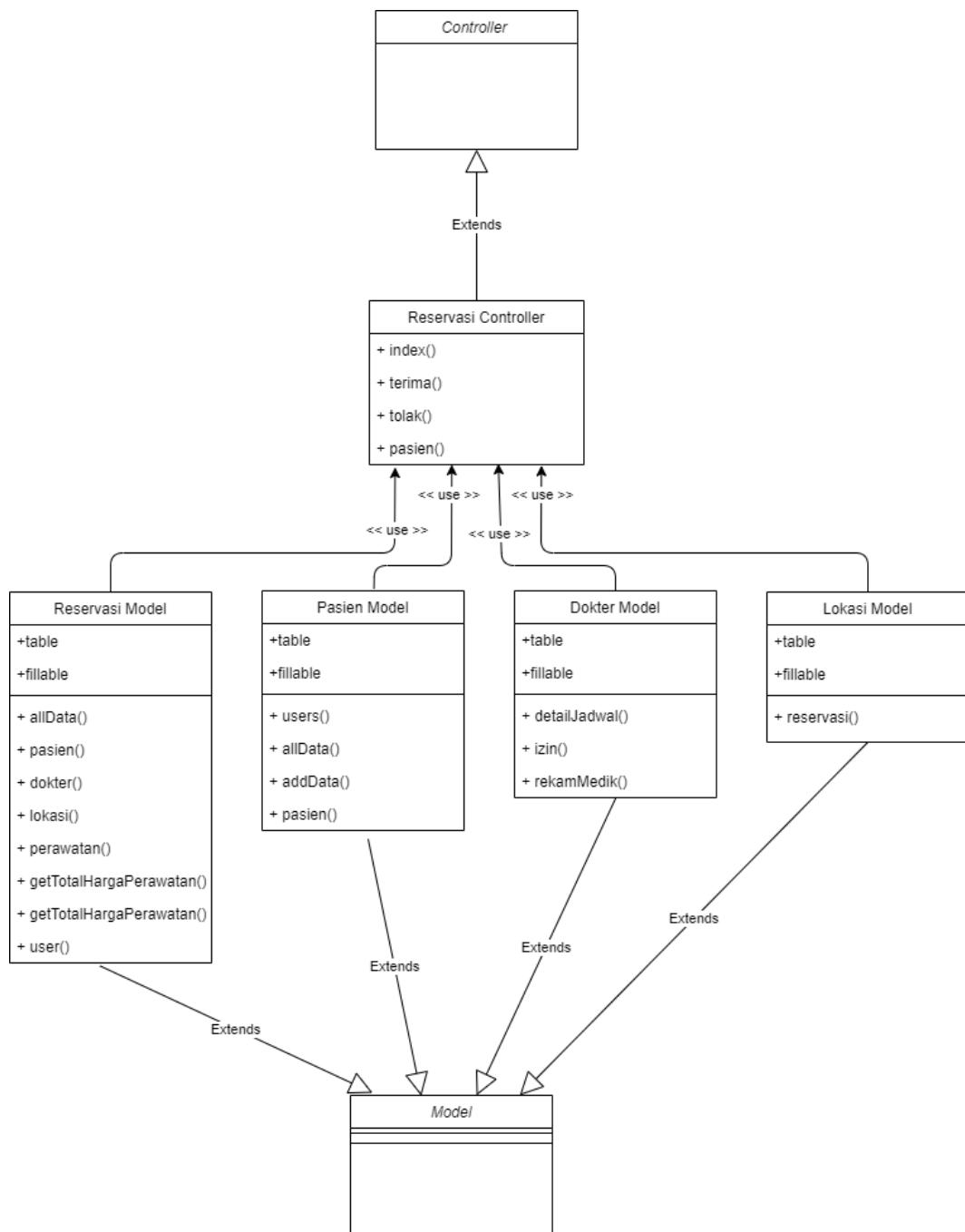
### 4.2.3 Class Diagram

*Class diagram* adalah salah satu jenis diagram dalam *Unified Modeling Language* (UML) yang digunakan untuk menggambarkan struktur dan hubungan antar kelas dalam suatu sistem. Class diagram membantu agar pembangunan sistem yang menggunakan paradigma *Object Oriented Programming* (OOP) menjadi lebih terstruktur, sehingga memudahkan proses pengembangan. Pada pembangunan *website* yang menggunakan *framework* Laravel maka akan terdiri dari kelas Controller dan kelas Model. Struktur kelas Controller terdiri dari fungsi-fungsi yang menjalankan fungsi tertentu, sementara struktur kelas Model terdiri dari atribut dan fungsi agar sistem bisa berinteraksi dengan database. Pada subbab ini akan dijelaskan class diagram yang terdapat pada sistem ini meliputi *class diagram* admin konfirmasi reservasi dari pasien, *class diagram* pengajuan permintaan izin dokter, serta *class diagram* pemilik melihat jumlah kunjungan pasien.

### 5.2.2.4 Class Diagram Konfirmasi Reservasi Dari Pasien

*Class diagram* admin konfirmasi reservasi dari pasien adalah *class diagram* yang digunakan dalam proses admin menerima atau menolak reservasi yang diajukan dari calon pasien. Pada class diagram ini terdapat 1 kelas reservasi *controller* yang memiliki fungsi index(), terima(), tolak(), pasien(). Pada diagram

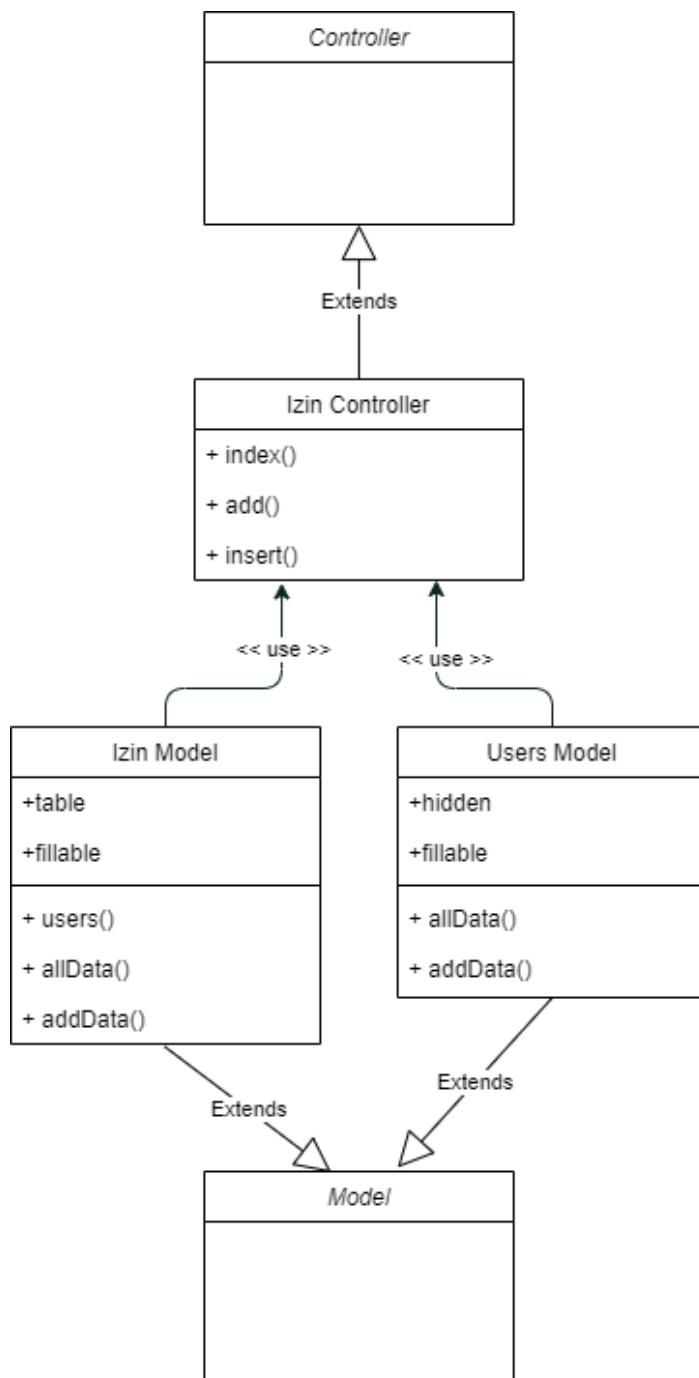
ini juga terdapat 4 kelas model diantaranya reservasi model, pasien model, dokter model, lokasi model. *Class diagram* ini dapat dilihat pada gambar 4.14



Gambar 4.14 Class Diagram Konfirmasi Reservasi Dari Pasien

#### 4.2.3.2 Class Diagram Pengajuan Permintaan Izin Dokter

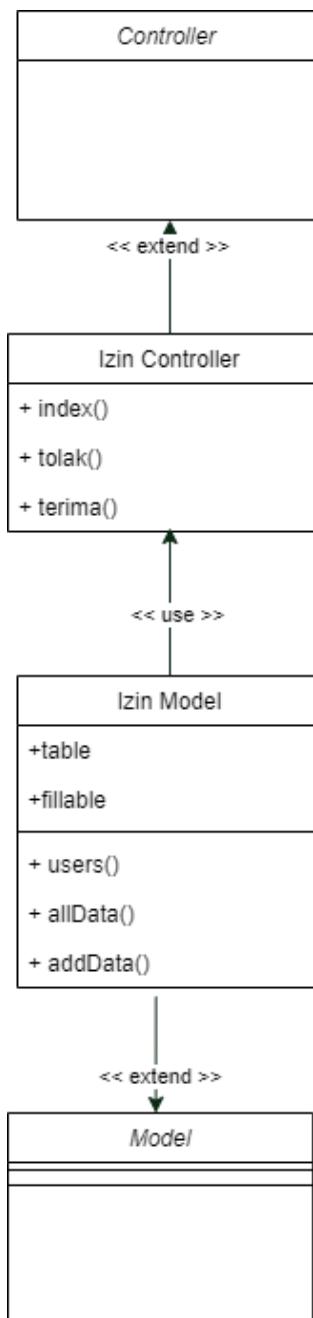
Class diagram pengajuan permintaan izin dokter adalah *class diagram* yang digunakan dalam proses permintaan izin dokter kepada *owner*. Pada diagram ini terdapat kelas izin dokter yang memiliki fungsi index(), add() dan insert(). *Class diagram* ini bisa dilihat pada gambar 4.15



Gambar 4.15 Class Diagram Pengajuan Permintaan Izin Dokter

#### 4.2.3.3 Class Diagram Pemilik Memutuskan Perizinan Dokter

Class diagram pemilik memutuskan perizinan dokter adalah *class diagram* yang digunakan dalam proses menolak atau menerima perizinan yang diajukan dokter kepada owner. Diagram ini memiliki kelas izin yang memiliki fungsi index(), tolak(), dan terima(). Diagram ini juga memiliki 1 kelas model yaitu izin model. Class diagram ini bisa dilihat pada gambar 4.16 berikut.



Gambar 4.16 Class Diagram Pemilik Memutuskan Perizinan Dokter

#### **4.2.4 Tampilan Mockup Antarmuka**

Mockup antarmuka merupakan bentuk visual desain antarmuka pengguna yang memberikan gambaran detail mengenai tata letak elemen-elemen yang ada di dalam suatu *website*. Dalam sub bab ini akan dijelaskan desain antarmuka dari form reservasi, tampilan data user, dashboard owner, dan landing page *website*.

##### **4.2.4.1 Mockup Antarmuka Halaman Form Reservasi**

Mockup antarmuka halaman form reservasi adalah halaman yang berfungsi untuk menerima inputan berupa informasi informasi yang dibutuhkan dalam melakukan reservasi. Form reservasi ini bisa diisi melalui akun admin dan pasien. Pada halaman ini user akan menginputkan tempat reservasi, dokter, jenis perawatan, jadwal perawatannya. Tampilan halaman ini dapat dilihat pada gambar 4.17 berikut.

The mockup shows a reservation form titled "Form Reservasi". At the top left is a "Kembali" button. The form fields include:

- A "Tempat" section with two options: "Padang" and "Bukittinggi".
- A "Nama Dokter" input field.
- A "Jenis Perawatan" dropdown menu.
- A "Jadwal" dropdown menu showing "09:00 - 18:00".
- A "Nomor HP" input field.
- A "Submit" button at the bottom left.

Gambar 4.17 Mockup Form Reservasi

#### 4.2.4.2 Mockup Antarmuka Halaman Data User

Mockup antarmuka halaman data user adalah halaman yang mana data user dikelola. Pada halaman ini terdapat list user *website* ini yang dapat dikelola oleh admin. Pada halaman ini admin dapat menghapus data, menambahkan, serta dapat mengedit data user yang ada pada klinik ini. Tampilan ini dapat dilihat pada gambar yang tertera pada gambar 4.18 berikut.

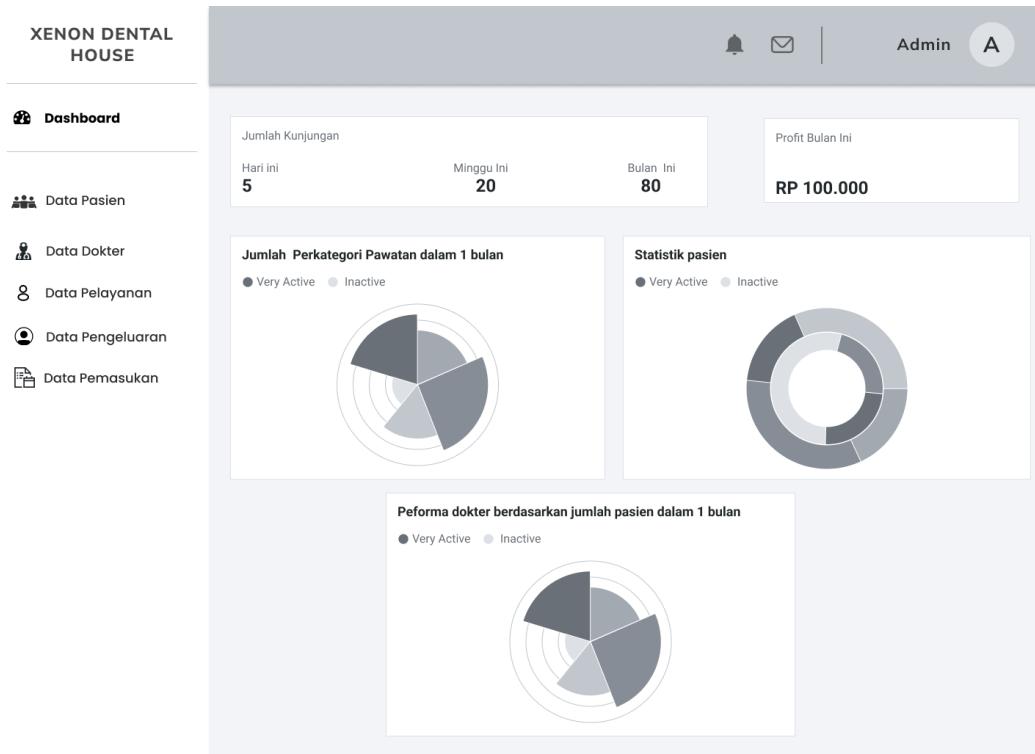
No	Nama	Email	Tipe	Aksi
1	Andi	<a href="mailto:andi@gmail.com">andi@gmail.com</a>	dokter	<button>Edit</button> <button>Hapus</button>
2	Budi	<a href="mailto:budi@gmail.com">budi@gmail.com</a>	pasien	<button>Edit</button> <button>Hapus</button>
3	Rauf	<a href="mailto:rauf@gmail.com">rauf@gmail.com</a>	admin	<button>Edit</button> <button>Hapus</button>
4	Fajri	<a href="mailto:fajri@gmail.com">fajri@gmail.com</a>	pasien	<button>Edit</button> <button>Hapus</button>
5	Arif	<a href="mailto:arif@gmail.com">arif@gmail.com</a>	pasien	<button>Edit</button> <button>Hapus</button>

Gambar 4.18 Mockup Tampillan data user

#### 4.2.4.3 Mockup Antarmuka Halaman Dashboard owner

Mockup antarmuka halaman dashboard adalah halaman yang menampilkan visualisasi data terhadap owner klinik Xenon Dental House. Data yang ditampilkan ini adalah data hasil analisis berupa jumlah pasien yang diterima dokter selama 1 bulan, pasien berdasarkan umur dari pasien, perawatan yang dilakukan pada pasien, serta jumlah kunjungan pasien. Pada halaman ini owner juga dapat melakukan pemilihan pada visualiasi untuk menampilkan bulan apa yang ingin

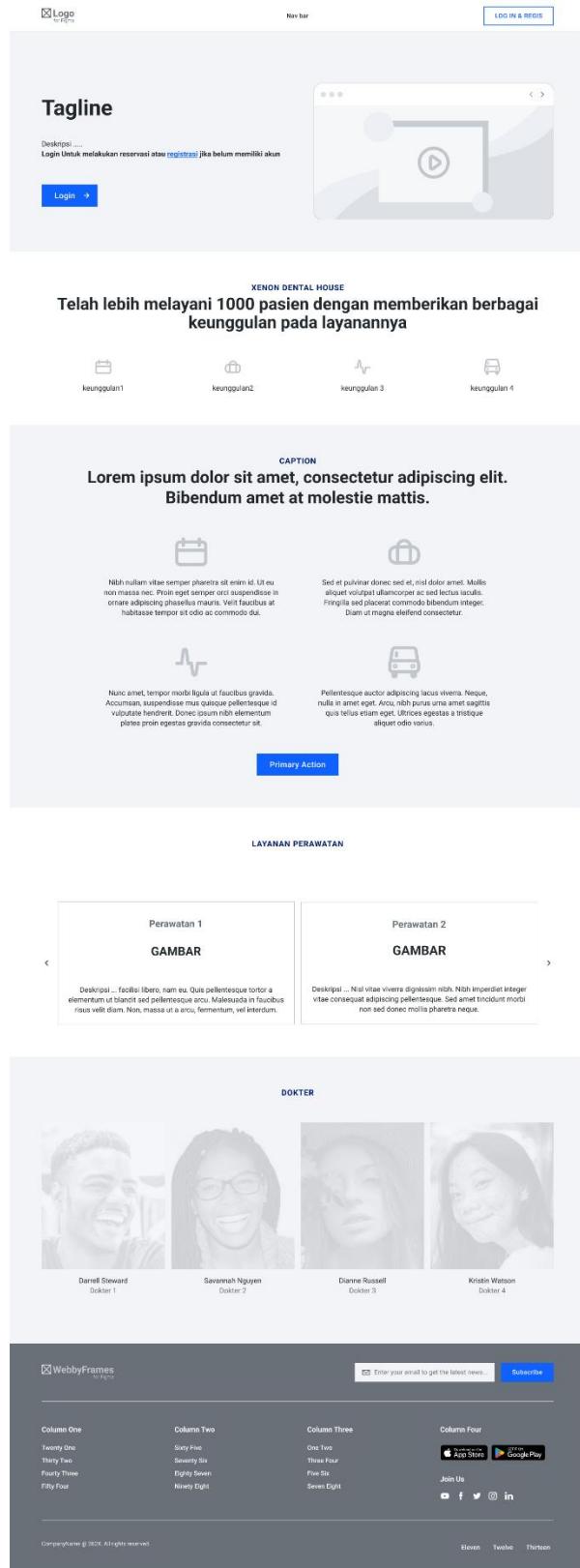
ditampilkan, sehingga owner dapat memfilter bulan apa yang ingin ditampilkan. Tampilan ini dapat dilihat pada gambar 4.19 berikut



Gambar 4.19 Mockup Dashboard akun owner

#### 4.2.4.4 Mockup Antarmuka Halaman Landing page

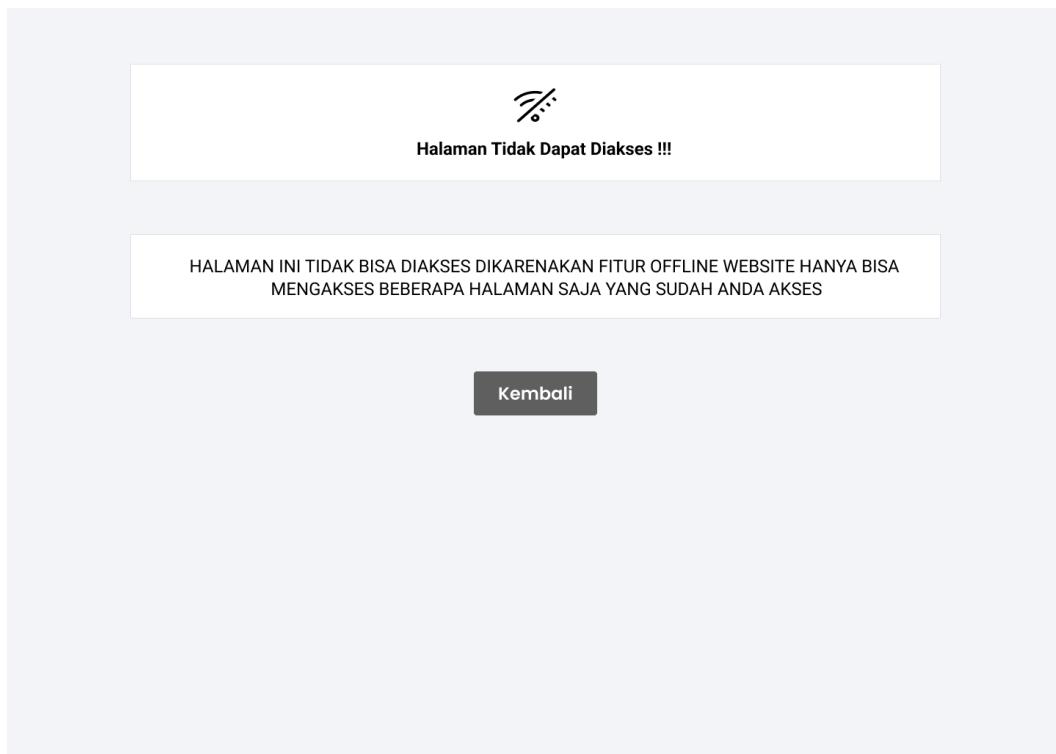
Mockup antarmuka halaman landing page adalah halaman yang menampilkan landing page dari *website* klinik gigi Xenon Dental House. Landing page adalah halaman tunggal yang dirancang untuk pemasaran dan branding dari *website* sehingga diharapkan dapat menarik pengunjung untuk keuntungan dari klinik seperti pada klinik ini yaitu melakukan reservasi untuk perawatan di klinik ini. Hasil dari mockup ini dapat dilihat pada gambar 4.20 berikut.



Gambar 4.20 Mockup Landing page website

#### **4.2.4.5 Mockup Antarmuka Offline PWA**

Mockup antarmuka offline PWA adalah halaman yang menampilkan tampilan ketika user mencoba mengakses website yang belum pernah diaksesnya setelah server mati. Salah satu keuntungan dari implementasi PWA adalah user bisa mengakses tampilan website meskipun server mati atau koneksi ke server terputus. Fitur offline ini berlaku ketika user sudah pernah mengakses url yang akan diakses ini sebelumnya, jika belum pernah maka tampilan ini yang akan ditampilkan oleh website. Mockup ini dapat dilihat pada gambar 4.21 berikut.



Gambar 4.20 Mockup Antarmuka Offline PWA

## **BAB V**

### **IMPLEMENTASI DAN PENGUJIAN**

#### **5.1 Implementasi Sistem**

Sistem informasi manajemen klinik Xenon Dental House dengan menggunakan teknologi *website* ini difungsikan sebagai sistem yang dapat mengelola dan menampilkan data dan informasi yang berkaitan dengan manajemen klinik dimulai dari reservasi, izin dokter, pengelolaan data layanan perawatan, hingga analisis data klinik. Pada sistem ini terdapat 4 aktor yang terlibat dalam 29 fungsional yang diharapkan dapat membantu terhadap semua proses pada sistem.

Implementasi *website* ini dilakukan dengan menggunakan perangkat keras dengan spesifikasi berikut:

1. Komputer dengan processor Intel CoreI i5-1035G1
2. Random Acces Memory (RAM) 8GB
3. Penyimpanan dengan kapasitas 476.94 GB

Spesifikasi perangkat lunak yang digunakan dalam implementasi sistem ini adalah sebagai berikut:

1. Sistem operasi windows 10
2. Web browser yang digunakan adalah google chrome versi
3. Code editor visual studio code
4. Webs server Built-in PHP Development Server
5. Laravel 11 dengan php 8.2
6. Database MySQL

##### **5.1.1 Pengkodean sistem**

Bagian ini menjelaskan tentang implementasi desain aplikasi dan arsitektur aplikasi yang telah dibahas pada bab sebelumnya dalam bentuk kode program. Beberapa potongan kode program yang akan diuraikan dalam bagian ini meliputi kode bagian route, controller, model dan view pada sistem ini.

### 5.1.1.1 Kode Program Route

Routing adalah proses yang mengarahkan permintaan HTTP ke bagian tertentu dari aplikasi. Dalam Laravel, routing berfungsi untuk memetakan URL yang diminta pengguna ke kontroler tertentu dalam aplikasi. Hal ini memungkinkan aplikasi web untuk merespons permintaan pengguna dengan cara yang sesuai, seperti menampilkan halaman, memproses request, atau mengelola data. Pada routing ini terdapat beberapa metode yang sering digunakan yaitu seperti metode get, post, delete. Pada dasarnya terdapat beberapa jenis route pada Laravel sebagai berikut.

1. Route dasar yang merupakan pemetaan url ke fungsi pengaan
2. Route dengan controller yang merupakan pemetaan url ke controller
3. Route group yang merupakan pemetaan kelompok yang memiliki kebutuhan middleware yang sama
4. Route nama yang merupakan route yang diberikan nama untuk referensi yang lebih mudah

Pada bagian ini akan dijelaskan 2 potongan kode route yang memiliki middleware yang berbeda yaitu role untuk user dokter dan route user admin yang dapat dilihat pada gambar 5.1 dan 5.2 berikut .

```
// Routes untuk dokter
Route::middleware(['role:Dokter'])->group(function () {
    Route::get('/dokter',[DokterHomeController::class, 'index'])->name('dokter.index');
    Route::get('/dataabsen',[IzinController::class, 'index'])->name('dataabsen.index');
    Route::get('/dataabsen/add',[IzinController::class, 'add'])->name('dataabsen.add');
    Route::get('/dataabsen/edit{id}',[IzinController::class, 'edit'])->name('dataabsen.edit');
    Route::post('/dataabsen/update{id}',[IzinController::class, 'update'])->name('dataabsen.update');
    Route::post('/dataabsen/insert',[IzinController::class, 'insert'])->name('dataabsen.insert');
    Route::delete('/dataabsen/destroy{id}',[IzinController::class, 'destroy'])->name('dataabsen.destroy');

    Route::get('/ddatapasien',[PasienController::class, 'index'])->name('ddatapasien.index');
    Route::get('/ddatapasien/add',[PasienController::class, 'add'])->name('ddatapasien.add');
    Route::post('/ddatapasien/insert',[PasienController::class, 'inser'])->name('ddatapasien.insert');
    Route::get('/ddatapasien/edit{id}',[PasienController::class, 'edit'])->name('ddatapasien.edit');
    Route::post('/ddatapasien/update{id}',[PasienController::class, 'update'])->name('ddatapasien.update');
    Route::delete('/ddatapasien/destroy{id}',[PasienController::class, 'destroy'])->name('ddatapasien.destroy');
    Route::get('/ddatapasien/rekam{id}',[RekamController::class, 'index'])->name('dokter.rekampasien');
    Route::get('/ddatapasien/rekam/edit{id}',[RekamController::class, 'edit'])->name('dokter.rekampasien.edit');
    Route::post('/ddatapasien/rekam/update{id}',[RekamController::class, 'update'])->name('dokter.rekampasien.update');
});
```

Gambar 5.1 Potongan Kode Route Dokter

```

: > web.php > Closure
Route::middleware(['role:admin'])->group(function () {
    // Routes untuk datadokter
    Route::get('/datadokter', [DataDokterController::class, 'index'])->name('datadokter.index');
    Route::get('/datadokter/add', [DataDokterController::class, 'add'])->name('datadokter.add');
    Route::post('/datadokter/insert', [DataDokterController::class, 'insert'])->name('datadokter.insert');
    Route::get('/datadokter/edit/{id}', [DataDokterController::class, 'edit'])->name('datadokter.edit');
    Route::post('/datadokter/update/{id}', [DataDokterController::class, 'update'])->name('datadokter.update');
    Route::delete('/datadokter/destroy/{id}', [DataDokterController::class, 'destroy'])->name('datadokter.destroy');

    // Routes untuk Jadwal DataDokter
    Route::get('/datadokter/jadwal/{id}', [JadwalController::class, 'index'])->name('dokterjadwal.index');
    Route::get('/datadokter/jadwal/add/{id}', [JadwalController::class, 'add'])->name('dokterjadwal.add');
    Route::post('/datadokter/jadwal/insert/{id}', [JadwalController::class, 'insert'])->name('dokterjadwal.insert');
    Route::get('/datadokter/jadwal/edit/{id}', [JadwalController::class, 'edit'])->name('dokterjadwal.edit');
    Route::post('/datadokter/jadwal/update/{id}', [JadwalController::class, 'update'])->name('dokterjadwal.update');
    Route::delete('/datadokter/jadwal/destroy/{id}', [JadwalController::class, 'destroy'])->name('dokterjadwal.destroy');

    // Routes untuk Reservasi Admin
    Route::get('/reservasi/admin/', [ReservasiController::class, 'index'])->name('reservasi.index');
    Route::get('/reservasi/admin/add', [ReservasiController::class, 'add'])->name('reservasi.add');
    Route::post('/reservasi/admin/insert', [ReservasiController::class, 'insert'])->name('reservasi.insert');
    Route::get('/reservasi/admin/edit/{id}', [ReservasiController::class, 'edit'])->name('reservasi.edit');
    Route::post('/reservasi/admin/update/{id}', [ReservasiController::class, 'update'])->name('reservasi.update');
    Route::delete('/reservasi/admin/destroy/{id}', [ReservasiController::class, 'destroy'])->name('reservasi.destroy');
    Route::post('/reservasi/admin/terima/{id}', [ReservasiController::class, 'terima'])->name('reservasi.terima');
    Route::post('/reservasi/admin/tolak/{id}', [ReservasiController::class, 'tolak'])->name('reservasi.tolak');
    Route::get('/reservasi/pasien', [ReservasiController::class, 'pasien'])->name('reservasi.pasien');
    Route::get('/api/dokter-by-lokasi', [ReservasiController::class, 'getDokterByLokasi']);
    Route::get('/api/unavailable-times', [ReservasiController::class, 'getUnavailableTimes']);
    Route::get('/api/available-dates', [ReservasiController::class, 'getAvailableDates']);
    Route::get('/api/available-days', [ReservasiController::class, 'getAvailableDays']);
    Route::get('/api/booked-times', [ReservasiController::class, 'getBookedTimes']);

    Route::get('/api/sesi-by-dokter-and-hari', [ReservasiController::class, 'getSesiByDokterAndHari']);
    Route::get('/reservasi/getDoctors', [ReservasiController::class, 'getDokterByLokasi'])->name('reservasi.getDoctors');
    Route::get('/reservasi/getTimeSlots', [ReservasiController::class, 'getTimeSlots'])->name('reservasi.getTimeSlots');
}

```

Gambar 5.2 Potongan Kode Route admin

### 5.1.1.2 Kode Program Controller

Controller adalah bagian yang bertanggung jawab untuk mengelola logika aplikasi dan merespons permintaan HTTP yang masuk. Controller menghubungkan rute (routing) dengan tampilan (views) serta berinteraksi dengan model untuk mengambil dan manipulasi data, dan bertindak sebagai pengatur alur aplikasi. Controller pada laravel pada umumnya memiliki metode standar seperti index() untuk menampilkan tampilan awal, create() untuk tampilan tempat penambahan data, store() untuk menambahkan data, show() untuk menampilkan detail dari tampilan index, edit() untuk tampilan tempat pengeditan data serta update() untuk memperbarui data yang sudah ada. Pada bagian ini akan dijelaskan controller untuk datadoktercontroller dan jadwalcontroller.

Datadoktercontroller adalah controller yang memproses data pada tabel dokter seperti data nama, alamat, nomor\_hp, dan lokasi\_id. Controller ini dapat menambahkan, menghapus, memperbarui serta dapat menghapus data dari tabel dokter. Sedangkan untuk jadwalcontroller adalah controller yang mengatur

proses yang dilakukan pada jadwal dokter. Controller ini juga dapat melakukan CRUD pada tabel detail jadwal yang memiliki atribut hari, sesi dan dokter id. Kode program ini dapat dilihat pada gambar 5.3 dan gambar 5.4 berikut.

```
class DataDokterController extends Controller

    public function add()
    {
        $lokasi = Lokasi::all();
        return view('admin.datadokter_add', compact('lokasi'));
    }

    public function insert(Request $request)
    {
        $request->validate([
            'nama' => 'required',
            'alamat' => 'required',
            'nomor_hp' => 'required',
            'lokasi_id'=> 'required',
        ], [
            'nama.required' => 'Nama harus diisi!',
            'alamat.required' => 'alamat harus diisi!',
            'nomor_hp.required' => 'nomor_hp harus diisi!',
            'lokasi_id.required' => 'lokasi praktek harus diisi'
        ]);
    }

    $data = [
        'nama' => $request->nama,
        'alamat' => $request->alamat,
        'nomor_hp' => $request->nomor_hp,
        'lokasi_id' => $request->lokasi_id
    ];

    $this->datadokter->addData($data);
    Alert::success('Berhasil!', 'Data dokter berhasil ditambahkan!');
    return redirect('/datadokter');
}

public function destroy($dokter_id)
{
    DB::table('dokter')->where('dokter_id', $dokter_id)->delete();

    // Menampilkan alert sukses dan mengarahkan kembali ke halaman data dokter
    Alert::success('Berhasil!', 'Data dokter berhasil dihapus!');
    return redirect('/datadokter');
}
```

Gambar 5.3 Potongan kode DataDokterController

```

class JadwalController extends Controller
{
    public function insert(Request $request, $id)
    {
        // Validasi input form
        $request->validate([
            'hari' => 'required',
            'sesi' => 'required',
        ], [
            'hari.required' => 'Hari harus diisi!',
            'sesi.required' => 'Sesi harus diisi!',
        ]);

        // Menyiapkan data yang akan disimpan
        $data = [
            'hari' => $request->hari,
            'sesi' => $request->sesi,
            'dokter_id' => $id,
        ];

        // Menyimpan data menggunakan model (asumsi ada method addData di model yang digunakan)
        $this->datadokter->addData($data);

        // Menampilkan pesan sukses menggunakan SweetAlert (asumsi Alert diimport dengan benar)
        Alert::success('Berhasil!', 'Data jadwal berhasil ditambahkan!');

        // Redirect ke halaman yang diinginkan dengan sintaks kurung keriting ganda
        return redirect("/datadokter/jadwal/{$id}");
    }

    public function destroy($jadwal_id)
    {
        // Hapus entri terkait di tabel detail_jadwal terlebih dahulu
        DB::table('detail_jadwal')->where('jadwal_id', $jadwal_id)->delete();

        // Menampilkan alert sukses dan mengarahkan kembali ke halaman data dokter
        Alert::success('Berhasil!', 'Data jadwal dokter berhasil dihapus!');
        $id_dokter = DB::table('detail_jadwal')->where('jadwal_id', $jadwal_id)->value('dokter_id');
        if($id_dokter==NULL){
            return redirect("/datadokter");
        }
        else{
            return redirect("/datadokter/jadwal/{$id_dokter}");
        }
    }
}

```

Gambar 5.4 Potongan kode DataDokterController

### **5.1.1.3 Kode Program Model**

Model adalah komponen dari arsitektur MVC (Model-View-Controller) yang bertugas untuk berinteraksi dengan basis data. Model merepresentasikan tabel dalam basis data dan menyediakan cara untuk mengakses serta memanipulasi data dalam tabel tersebut. Pada bagian ini akan dijelaskan 2 model yaitu model detail jadwal dan model dokter. Pada model ini terdapat atribut khusus seperti \$table yang merepresentasikan nama tabel dari database, \$primary\_key yang merepresentasikan primary key dari tabel dan \$fillable yang merupakan atribut dari tabel yang bisa diisi dengan data.

Model detailjadwal adalah model yang merepresentasikan tabel jadwal dokter pada database yang memiliki primary key jadwal\_id serta memiliki atribut lain seperti dokter\_id, hari dan sesi. Model ini memiliki hubungan many to one pada tabel dokter yang dapat dilihat pada fungsi dokter(). Model ini juga memiliki fungsi allData yang mana akan melakukan request semua data jadwal ke database dan memiliki fungsi addData yang akan menambahkan data ke tabel detail jadwal pada database. Kode program dari model ini dapat dilihat pada gambar 5.5.

Model dokter adalah model yang merepresentasikan tabel dokter pada database yang memiliki primary key dokter\_id serta memiliki atribut lain seperti nama, alamat, nomor\_hp, lokasi\_id dan user\_id. Model ini memiliki hubungan one to many pada tabel detail jadwal yang dapat dilihat pada fungsi detail jadwal dan memiliki fungsi izin yang memperlihatkan hubungan hasmany kepada tabel izin. Model ini juga memiliki fungsi allData yang mana akan melakukan request semua data dokter ke database dan memiliki fungsi addData yang akan menambahkan data ke tabel dokter. Kode program dari model ini dapat dilihat pada gambar 5.6 dibawah ini.

```
class DetailJadwal extends Model
{
    use HasFactory;

    protected $table = 'detail_jadwal';
    protected $primaryKey = 'jadwal_id';
    public $timestamps = false;

    protected $fillable = [
        'dokter_id',
        'hari',
        'sesi'
    ];

    public function dokter()
    {
        return $this->belongsTo(Dokter::class, 'dokter_id');
    }

    public function allData()
    {
        return DB::table('detail_jadwal')->get();
    }

    public function addData($data)
    {
        DB::table('detail_jadwal')->insert($data);
    }
}
```

Gambar 5.5 Potongan kode model detail jadwal

```

class Dokter extends Model
{
    use HasFactory;

    protected $table = 'dokter';
    protected $primaryKey = 'dokter_id';
    public $timestamps = false;

    protected $fillable = [
        'nama',
        'alamat',
        'nomor_hp',
        'lokasi_id',
        'user_id'
    ];

    public function detailJadwal()
    {
        return $this->hasMany(DetailJadwal::class, 'dokter_id');
    }

    public function izin()
    {
        return $this->hasMany(Izin::class, 'dokter_id');
    }

    public function rekamMedik()
    {
        return $this->hasMany(RekamMedik::class, 'dokter_id');
    }
}

```

Gambar 5.6 Potongan kode model dokter

#### 5.1.1.4 Kode Program View

View adalah komponen dari arsitektur MVC (Model-View-Controller) yang bertanggung jawab untuk menampilkan data kepada pengguna. View adalah tempat di mana menentukan bagaimana data yang dikirim dari controller ke pengguna akhir akan ditampilkan. Pada bagian ini akan dijelaskan 2 view yaitu view datadokter\_add dan datadokter yang menggunakan template engine bawaan Laravel yaitu file blade. View datadokter\_add adalah view yang menampilkan form saat akan ditambahkannya data dokter baru oleh admin. Pada view ini admin akan

diminta mengisikan nama, alamat, nomor\_hp dan lokasi\_id dari dokter. View datadokter adalah view yang menampilkan data dokter setelah ditambahkannya data dokter oleh admin. View ini menampilkan no,nama, dan alamat dari dokter. View ini juga menyediakan tombol dan link yang akan mengarahkan admin ke tampilan detail jadwal dari dokter, menghapus data dokter, menambahkan serta mengedit data dokter. Kedua view ini dapat dilihat pada gambar 5.7 dan gambar 5.8 berikut.

```

@extends('layout.v_template')

@section('main-content')
<h3 class="font-weight-bold mx-4">Tambah data Dokter </h3>
<!-- Page Heading -->
<div class="mx-4">
    <form action="/datadokter/insert" method="post" enctype="multipart/form-data">
        @csrf
        <div class="form-group">
            <label>Nama Dokter</label>
            <input name="nama" class="form-control @error('nama') is-invalid @enderror" value="{{ old('nama') }}>
            <div class="invalid-feedback">
                @error('nama')
                    {{ $message }}
                @enderror
            </div>
        </div>

        <div class="form-group">
            <label>Alamat</label>
            <input name="alamat" class="form-control @error('alamat') is-invalid @enderror" value="{{ old('alamat') }}>
            <div class="invalid-feedback">
                @error('alamat')
                    {{ $message }}
                @enderror
            </div>
        </div>

        <div class="form-group">
            <label>Nomor Hp </label>
            <input name="nomor_hp" class="form-control @error('nomor_hp') is-invalid @enderror" value="{{ old('nomor_hp') }}>
            <div class="invalid-feedback">
                @error('nomor_hp')
                    {{ $message }}
                @enderror
            </div>
        </div>
    </form>
</div>

```

Gambar 5.7 Potongan kode view datadokter\_add

```

@extends('layout.v_template')

@section('main-content')
<!-- Page Heading -->



### Data Dokter </h3> | no | Nama | No HP | Jadwal | Aksi | |-------------------------|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------| | {{ \$loop->iteration }} | {{ \$dok->nama }} | @php \$nomor_hp = \$dok->nomor_hp; if (substr(\$nomor_hp, 0, 2) == '08') { \$nomor_hp = '628' . substr(\$nomor_hp, 2); } @endphp {{ \$nomor_hp }} | <a href="/datadokter/jadwal/{{ \$dok-&gt;dokter_id }}">Detail Jadwal</a> <div class="d-flex"> <a class="btn btn-sm btn-primary mr-2" href="/datadokter/edit/{{ \$dok-&gt;dokter_id }}">Edit</a> <form action="{{ route('datadokter.destroy', [\$dok-&gt;dokter_id]) }}" button="" class="btn btn-danger" display="inline" method="POST" onsubmit="return confirm('Yakin ingin menghapus? Data yang berhubungan dengan data dokter ini : {{ csrf() }} @method(['DELETE'])" type="submit">Hapus </form> </div> | <a href="/datadokter/jadwal/{{ \$dok-&gt;dokter_id }}">Detail Jadwal</a> <div class="d-flex"> <a class="btn btn-sm btn-primary mr-2" href="/datadokter/edit/{{ \$dok-&gt;dokter_id }}">Edit</a> </div> |


```

Gambar 5.8 Potongan kode View datadokter

#### 5.1.1.5 Kode Program Service Worker dan Manifest untuk PWA

Dalam penerapan mode offline PWA pada laravel diperlukan 2 file khusus yang bernama service\_worker.js dan manifest.json. Service worker adalah file JavaScript yang berjalan di background browser, terpisah dari halaman web. Ini bertindak sebagai perantara aplikasi web dan jaringan yang memungkinkan *website* untuk melakukan *caching* konten, pengelolaan permintaan jaringan dan akses *offline website*. Manifest file adalah file yang memberikan informasi tentang aplikasi agar bisa diinstall seperti aplikasi native dengan ikon, nama, warna tema, dan pengaturan tampilan. Pada manifest terdapat atribut seperti nama aplikasi ketika diinstall,nama singkat, url awalnya, display dan background. Potongan kode dari service worker dan anifest bisa dilihat pada gambar 5.9 dan gambar 5.10 berikut.

```

self.addEventListener('install', event => {
  event.waitUntil(
    caches.open(CACHE_NAME)
      .then(cache => {
        return cache.addAll(urlsToCache);
      })
  );
});

self.addEventListener('fetch', event => {
  event.respondWith(
    fetch(event.request).then(response => {
      // Jika berhasil mengambil dari jaringan, simpan di cache dan kembalikan respons
      return caches.open(CACHE_NAME).then(cache => {
        cache.put(event.request, response.clone());
        return response;
      });
    }).catch(() => {
      // Jika jaringan gagal, coba ambil dari cache
      return caches.match(event.request).then(response => {
        return response || caches.match('/offline');
      });
    })
  );
});

```

Gambar 5.9 Kode serviceworker.js

```
{
  "name": "Your App Name",
  "short_name": "AppName",
  "start_url": "/",
  "display": "standalone",
  "background_color": "#ffffff",
  "theme_color": "#000000",
  "icons": [
    {
      "src": "/images/icons/icon-72x72.png",
      "sizes": "72x72",
      "type": "image/png"
    },
    {
      "src": "/images/icons/icon-96x96.png",
      "sizes": "96x96",
      "type": "image/png"
    }
  ]
}
```

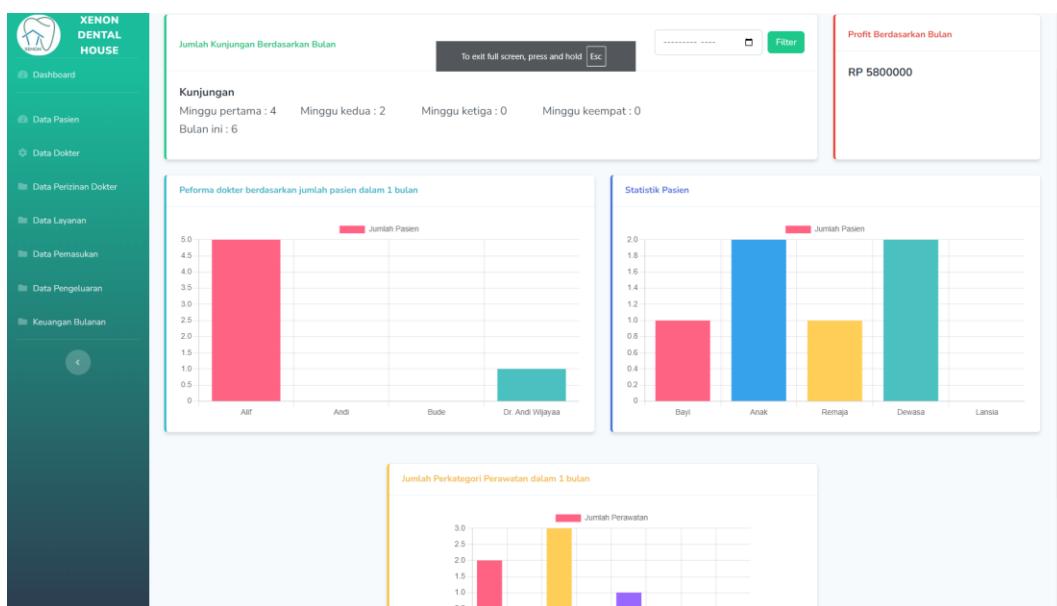
Gambar 5.10 Kode manifest.json

## 5.1.2 Implementasi Antarmuka sistem

Implementasi antarmuka sistem informasi manajemen klinik ini dilakukan menggunakan template engine Laravel blade yang berisikan HTML(HyperText Markup Language) , CCS(Cascading style sheets) serta javascript. Pada bagian ini akan dijelaskan implementasi antarmuka untuk tampilan dashboard owner, dashboard admin, data reservasi dari admin dan data reservasi dari pasien. Implementasi antarmuka lainnya dapat dilihat pada lampiran.

### 5.1.2.1 Implementasi Antarmuka Halaman Dashboard Owner

Halaman dashboard owner adalah halaman yang menyediakan visualisasi dari data yang ada di klinik. Halaman ini menyediakan informasi profit klinik, visualisasi berupa jumlah kunjungan dalam satu bulan serta terdapat jumlah kunjungan perminggunya pada bulan itu. Disamping itu juga terdapat visualisasi bar chart performa dokter berdasarkan jumlah kunjungan yang diterima oleh dokter , visualisasi jumlah dokter berdasarkan kategori umurnya dan visualisasi jumlah per kategori perawatan yang dilakukan sebanyak perawatan yang sudah dilakukan. Data data yang ditampilkan ini juga bisa ditampilkan berdasarkan bulan yang ingin kita pilih. Hasil dari implementasi halaman dashboard owner ini dapat dilihat pada gambar 5.11 berikut.



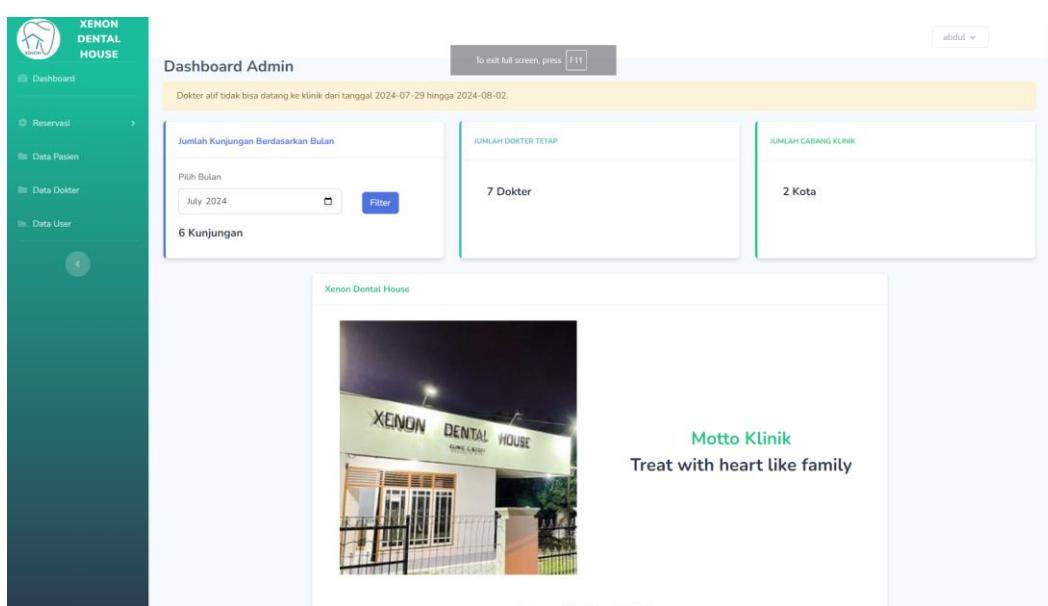
Gambar 5.11 Implementasi Antarmuka Halaman Dashboard Owner

### 5.1.2.2 Implementasi Antarmuka Halaman Dashboard Admin

Halaman dashboard admin adalah halaman pertama yang akan ditampilkan setelah admin login. Halaman ini berisikan informasi terkait jumlah kunjungan pasien dalam bulan ini, jumlah dokter yang bekerja di klinik gigi dan jumlah cabang yang dimiliki oleh klinik gigi. Pada halaman ini juga ditampilkan motto dari klinik gigi xenon dental house. Data data ini juga bisa difilter berdasarkan bulan bulan yang ada sesuai dengan keinginan admin. Hasil dari implementasi ini dapat dilihat pada gambar 5.12 berikut.

### 5.1.2.3 Implementasi Antarmuka Halaman Data Reservasi Admin

Halaman data reservasi admin adalah halaman yang berisikan data inputan reservasi yang telah berhasil dilakukan oleh admin atau data reservasi yang diajukan oleh pasien dan telah disetujui oleh admin. Pada halaman ini ditampilkan data reservasi dimulai dari nama pasien, lokasi perawatan, nama dokter yang akan melakukan praktek, tanggal, nomor hp, jam dan rekam medik dari pasien. Dari halaman ini admin bisa pindah ke berbagai halaman yang diinginkan seperti halaman tambah, edit data reservasi, halaman rekam medik, serta dapat menghapus data reservasi jika diperlukan. Hasil implementasi ini dapat dilihat pada gambar 5.13 berikut.



Gambar 5.12 Implementasi Antarmuka Dashboard Admin

No	Nama Pasien	Lokasi Perawatan	Nama Dokter	Tanggal	Nomor HP	Jam	Rekam Medik	Aksi
1	Budi Santoso	Padang	Alif	2024-10-14	6289617702747	10:00:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
2	Budi Santoso	Padang	Alif	2024-10-07	6289617702747	10:00:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
3	alif abdul	Padang	Alif	2024-10-07	6282288051901	10:00:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
4	dimas	Padang	Alif	2024-10-07	6282288051901	13:30:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
5	tono	Padang	Dr. Andi Wijayaa	2024-08-05	6282288051901	10:00:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
6	dimas	Padang	Dr. Andi Wijayaa	2024-08-01	6282288051901	10:00:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
7	Robert	Padang	Alif	2024-07-29	6285348923	10:00:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
8	Siti Aminah	Padang	Alif	2024-07-15	6282288051901	11:30:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>

Showing 1 to 8 of 8 entries

[Tambah Reservasi Pasien Baru](#) [Tambah Reservasi Pasien Lama](#)

Copyright © Xenon Dental House

Gambar 5.13 Implementasi Antarmuka Reservasi dari admin

#### 5.1.2.4 Implementasi Antarmuka Halaman Data Reservasi dari Pasien

Halaman ini merupakan halaman yang menampilkan data reservasi yang diajukan oleh pasien. Data ini berisikan data nama pasien, lokasi, nama dokter, tanggal dan jam. Pada halaman ini admin dapat melakukan penolakan atau penerimaan pada reservasi yang diajukan pasien. Hasil dari implementasi ini dapat dilihat pada gambar 5.14 berikut

No	Nama Pasien	Lokasi Perawatan	Nama Dokter	Tanggal	Jam	Aksi
1	Robert	Padang	Alif	2024-07-29	10:00:00	<button>Terima</button> <button>Tolak</button>

Showing 1 to 1 of 1 entries

[Previous](#) [1](#) [Next](#)

Copyright © Xenon Dental House

Gambar 5.14 Implementasi Antarmuka Reservasi Pasien

## 5.2 Pengujian Sistem

Pengujian sistem bertujuan untuk memastikan bahwa aplikasi yang dikembangkan sesuai dengan desain sistem yang telah direncanakan, memastikan hasil yang dirproses dari sistem sama dengan hasil yang diharapkan. Pengujian aplikasi dilakukan menggunakan metode black box. Pengujian sistem dengan metode black box dilakukan dengan membandingkan input dan output yang diharapkan pada setiap fungsionalitas tanpa memperhatikan proses internal dan kinerja aplikasi. Pengujian dilakukan pada semua aktor yang terlibat pada sistem yaitu admin, dokter, owner dan pasien yang menghasilkan hasil pengujian. Pengujian ini lebih lengkapnya dapat dilihat pada lampiran.

### 5.2.1 Fokus Pengujian

Pengujian aplikasi ini berfokus pada penggunaan data uji yang diambil dari aplikasi yang telah dikembangkan. Dalam pengujian ini, terdapat dua puluh lima item yang akan dievaluasi. Fokus pengujian dapat dilihat pada tabel 5.1 dibawah ini.

Tabel 5.1 Pengujian Aplikasi

No	Item Uji	Jenis Aplikasi	Pengguna	Detail Pengujian
1	Authentikasi dan Pengelolaan Akun	Website	Semua User	Login, ganti password, logout, ganti informasi profil, menghapus akun pribadi
2	Lihat Dasboard	Website	Admin, Owner, Dokter	Lihat, Filter
3	Konfirmasi reservasi dari Pasien	Website	Admin	Lihat, Tolak, terima
4	Reservasi pasien	Website	Admin	Lihat, tambah, edit, hapus
5	Mengelola data pasien	Website	Admin	Lihat, edit, hapus

Tabel 5.1 Pengujian Aplikasi(lanjutan)

6	Mengelola data rekam medik	Website	Admin	Lihat, edit
7	Mengelola data dokter, jadwal dokter	Website	Admin	Lihat, tambah, edit, hapus
8	Mengelola data user	Website	Admin	Lihat, tambah, edit, hapus
9	Registrasi Akun	Website	Pasien	Registrasi akun baru
10	Reservasi klinik	Website	Pasien	Tambah, Edit
11	Melihat Riwayat reservasi	Website	Pasien	Lihat
12	Lihat Jadwal Praktek	Website	Dokter	Lihat
13	Mengelola data pasien	Website	Dokter	Lihat, edit, hapus
14	Mengajukan permintaan izin	Website	Dokter	Lihat, tambah, edit, hapus
15	Melihat data profit klinik	Website	Owner	Lihat, Filter
16	Melihat jumlah kunjungan pasien	Website	Owner	Lihat, Filter
17	Melihat jumlah penanganan perawatan berdasarkan kategori	Website	Owner	Lihat, Filter
18	Melihat performa dokter berdasarkan jumlah pasien	Website	Owner	Lihat, Filter
19	Melihat data pasien berdasarkan umur	Website	Owner	Lihat, Filter
20	Melihat Data dokter	Website	Owner	Lihat
21	Melihat Data Pasien	Website	Owner	Lihat
22	Melihat Data Pemasukan, keuangan	Website	Owner	Lihat
23	Memutuskan perizinan dokter	Website	Owner	Terima, Tolak

Tabel 5.1 Pengujian Aplikasi(lanjutan)

24	Mengelola pencatatan pengeluaran klinik	<i>Website</i>	Owner	Lihat, tambah, edit, hapus
25	Mengelola harga perawatan	<i>Website</i>	Owner	Lihat, tambah, edit, hapus

### 5.2.2 Kasus Hasil Pengujian

Kasus hasil pengujian adalah pembahasan mengenai kasus dan hasil pengujian terhadap fungsionalitas aplikasi yang telah dilakukan berdasarkan fokus pengujian yang tercantum pada Tabel 5.1. Dalam sub bab ini akan dijelaskan mengenai kasus hasil pengujian yang mencakup Input Reservasi dari Admin, Registrasi Akun Pasien, Admin menerima pengajuan reservasi dari pasien

#### 5.2.2.1 Input Reservasi dari Admin

Pada pengujian ini admin akan menginputkan reservasi ketika pasien datang klinik Xenon Dental house. Pada kasus kali ini pasien bernama Andy meminta reservasi pada tanggal 18 Juli 2024. Hasil dari pengujian ini dapat dilihat pada tabel 5.2 dibawah ini :

Tabel 5.2 Pengujian kondisi benar input reservasi dari admin

<b>Kasus dan Hasil Uji (benar)</b>	
Data masukan	Semua data yang dibutuhkan untuk reservasi seperti detail reservasi dan data pribadi pasien
Hasil yang diharapkan	Data tersimpan dan system menampilkan list reservasi yang sudah ditambahkan
Pengamatan	Sistem berhasil menyimpan dan menampilkan list reservasi yang sudah ditambahkan beserta notifikasi berhasil
Hasil	Sesuai

Pengujian diawali dengan admin membuka menu reservasi admin pada website. Setelah berada pada halaman reservasi, admin menekan tombol tambahkan reservasi. Admin akan diarahkan pada halaman form tambah reservasi. Admin perlu mengisi semua form yang diperlukan untuk reservasi, setelah mengisi semua form maka admin menekan tombol submit. Jika berhasil akan muncul notifikasi berhasil dan admin akan diarahkan kepada halaman list reservasi. Data yang baru diinputkan akan muncul pada list reservasi. Pengujian ini dapat dilihat ada gambar 5.15, 5.16, dan 5.17 berikut.

No	Nama Pasien	Lokasi Perawatan	Nama Dokter	Tanggal	Nomor HP	Jam	Rekam Medik	Aksi
1	Budi Santoso	Padang	Alif	2024-10-14	6289617702747	10:00:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
2	Budi Santoso	Padang	Alif	2024-10-07	6289617702747	10:00:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
3	alif abdul	Padang	Alif	2024-10-07	6282288051901	10:00:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
4	dimas	Padang	Alif	2024-10-07	6282288051901	13:30:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
5	tono	Padang	Dr. Andi Wijayaa	2024-08-05	6282288051901	10:00:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
6	dimas	Padang	Dr. Andi Wijayaa	2024-08-01	6282288051901	10:00:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
7	Robert	Padang	Alif	2024-07-29	6285348923	10:00:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
8	Siti Aminah	Padang	Alif	2024-07-15	6282288051901	11:30:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>

Gambar 5.15 Tampilan awal reservasi dari admin

Gambar 5.16 Tampilan Form tambah reservasi

No	Nama Pasien	Lokasi Perawatan	Nama Dokter	Tanggal	Nomor HP	Jam	Rekam Medik	Aksi
1	Anisa	Padang	Dr. Andi Wijayaa	2024-07-18	62822050022	11:00:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
2	Andy	Padang	Dr. Andi Wijayaa	2024-07-18	62822881222	13:00:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
3	Siti Aminah	Padang	Alif	2024-07-15	628961770247	11:30:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
4	Budi	Padang	Alif	2024-07-09	628654345676543	17:00:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>
5	Badu	Padang	Alif	2024-07-01	628545215621	10:00:00	Rekam Medik	<button>Edit</button> <button>Hapus</button>

Showing 1 to 5 of 5 entries

[Tambahkan Reservasi](#)

Copyright © Xenon Dental House

Gambar 5.17 Tampilan reservasi setelah data berhasil ditambahkan

Selanjutnya adalah pengujian alternatif, pengujian alternatif adalah pengujian untuk kasus yang berbeda dibandingkan kasus benar. Pengujian alternatif untuk fungsional input reservasi terjadi ketika admin tidak mengisi semua form yang ada di reservasi. Hasil pengujian ini dapat dilihat pada tabel 5.3

Tabel 5.3 Pengujian alternatif reservasi dari admin

<b>Kasus dan Hasil Uji (alternatif)</b>	
Data masukan	Mengisikan hanya dari form reservasi
Hasil yang diharapkan	Akan muncul notifikasi untuk mengisi form yang dikosongkan
Pengamatan	Notifikasi form yang dikosongkan menunjukkan untuk mengisikan form yang kosong
Hasil	Sesuai

The screenshot shows a web application interface for 'XENON DENTAL HOUSE'. On the left, there's a sidebar with navigation links: Dashboard, Reservasi (selected), Data Pasien, Data Dokter, and Data User. The main content area is titled 'Form Tambah Reservasi' (Add Reservation Form). It contains several input fields: 'Nama' (Name) with value 'Hafiz', 'Tempat Lahir' (Birthplace) with value 'Bukittinggi', 'Tanggal Lahir' (Birthdate) with value '10/07/2024', 'Pekerjaan' (Occupation) with an empty input, 'Alamat' (Address) with an empty input and a validation message 'Please fill out this field.', 'No Telepon' (Phone Number) with an empty input, 'Perawatan (Pilih maksimal 2)' (Treatment (Select up to 2)) with an empty input, and 'Lokasi' (Location) with a dropdown menu showing 'Pilih Lokasi'. At the top right, there's a user dropdown labeled 'abdul.'

Gambar 5.18 Tampilan notifikasi saat input form reservasi tidak diisi

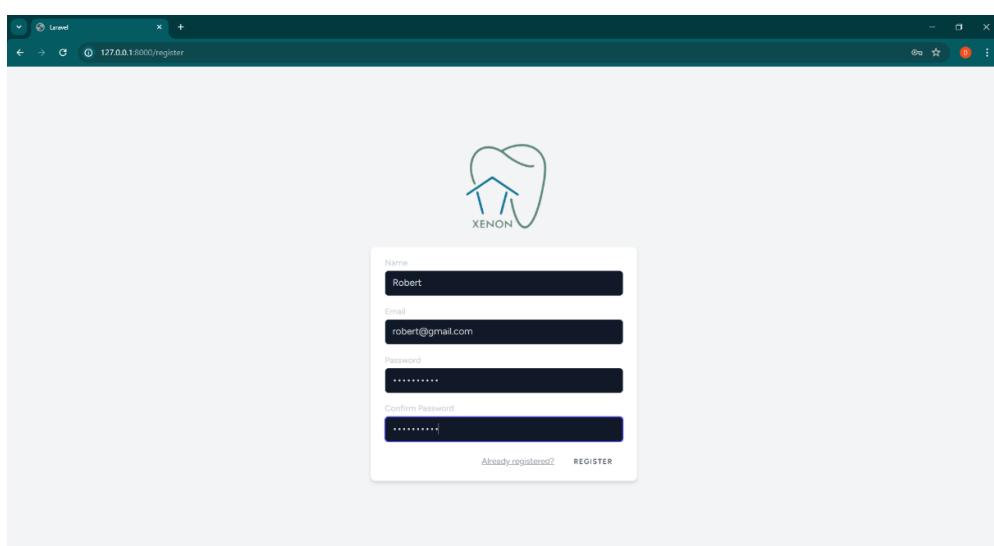
### 5.2.2.2 Registrasi Akun pasien

Pada pengujian ini pasien akan melakukan registrasi akun baru agar bisa melakukan reservasi pada *website*. Pada kasus kali ini pasien melakukan registrasi dengan mengisi pada halaman registrasi. Hasil dari pengujian ini dapat dilihat pada tabel 5.4 dibawah ini :

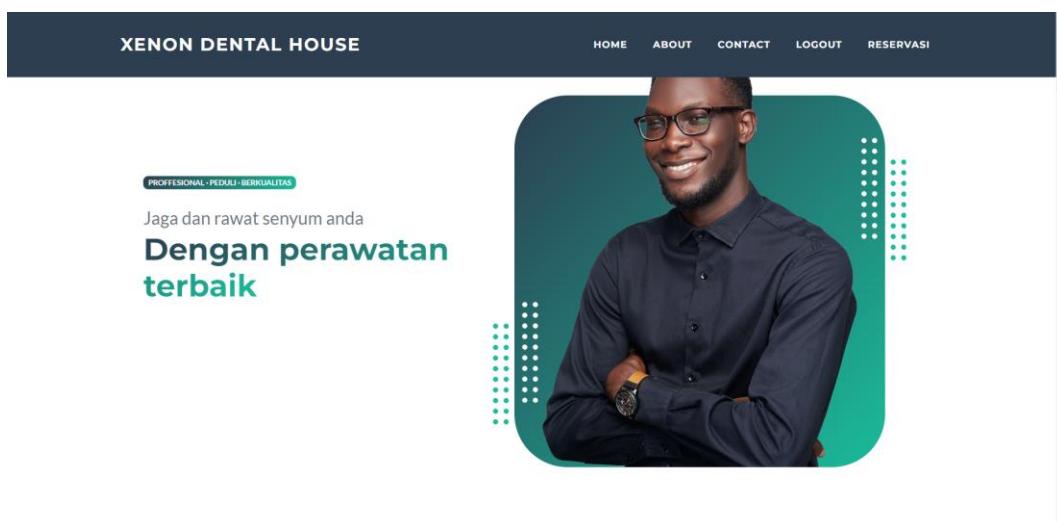
Tabel 5.4 Pengujian kondisi benar registrasi pasien

<b>Kasus dan Hasil Uji (benar)</b>	
Data masukan	Semua data yang dibutuhkan untuk registrasi, seperti nama, email, password
Hasil yang diharapkan	Data tersimpan dan system akan menampilkan landing page untuk akun pasien yang sudah login
Pengamatan	Sistem berhasil menyimpan data dan berhasil mengarahkan ke halaman landing page
Hasil	Sesuai

Pengujian diawali dengan pasien masuk pada halaman registrasi akun pada *website*. Setelah berada pada halaman registrasi, pasien mengisikan data data yang diperlukan untuk membuat akun baru. Setelah mengisi data data tersebut pasien menekan tombol register. Ketika berhasil maka pasien akan diarahkan ke tampilan landing page serta pasien bisa mengakses menu reservasi pada landing page. Data user akan terlihat bertambah di tabel user database *website*. Hasil pengujian ini dapat dilihat pada gambar 5.19, 5.20, dan 5.21.



Gambar 5.19 Tampilan form registrasi akun



Gambar 5.20 Tampilan Setelah pasien berhasil register

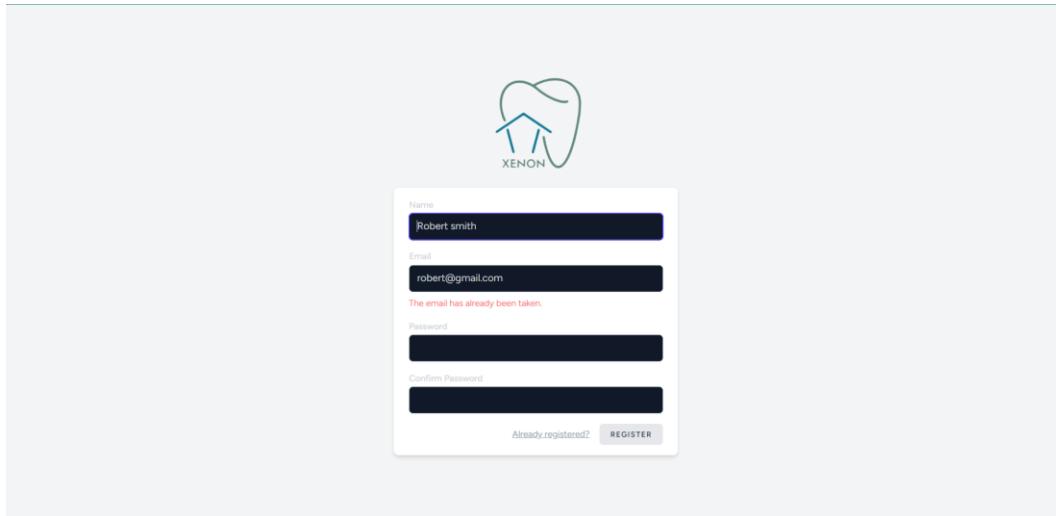
Database: dixip_logistics » Table: users										
Browse		Structure		SQL		Search		Operations		
<b>Showing rows 0-5 (total: 6). Query took 0.0003 seconds.</b>										
<b>SELECT *</b> FROM `users`										
<input type="checkbox"/> Profiling	<input type="checkbox"/> Edit inline	<input type="checkbox"/> [Edit]	<input type="checkbox"/> Explain SQL	<input type="checkbox"/> Create PHP code	<input type="checkbox"/> Refresh					
<input type="checkbox"/> Show all	Number of rows:	25	Filter rows:	Search this table	Sort by key:	None				
<b>Extra options</b>										
#	id	name	email	role	email_verified_at	password	remember_token	created_at	updated_at	
1	1	alfadilraufi@gmail.com	admin	NULL	\$2y\$12\$yHZE6lyQqvMsiAEPze/ydg62zCDku4dnZUTW6...	zm67HUu6Bw10VjQB7qvkhHlPq7h3QLMfl8jzakmH...	2024-05-14 11:08:17	2024-05-14 11:08:17		
2	2	alif	abdu@gmail.com	admin	NULL	\$2y\$12\$uJua7f0pmmyhEgYup5pAgJhBdvQab...99	QRgvHu2fRmvpqby5DTDwewEc4IRd3h5QOsh4npqASkeW...	2024-06-11 13:30:08	2024-06-11 13:30:08	
3	3	raf	rauf@mail.com	owner	NULL	\$2y\$12\$jeutlaTHPRyPmc4BgIgD7pJh5Lbcq4S...	NULL	2024-06-16 15:23:51	2024-06-16 15:23:51	
4	4	abdu	4abdu@gmail.com	Dokter	NULL	\$2y\$12\$9pJUG7YU1LcS9fIzaGK_80xRz5R2mUN...	NULL	NULL	NULL	
5	5	bob	bob@bob.com	Pasien	NULL	\$2y\$12\$QOHW6n89zKjMVnWWCCf1oddzhVay0V7NXXqg...	NULL	2024-07-03 16:10:31	2024-07-03 16:10:31	
6	6	alif	alif@gmail.com	Pasien	NULL	\$2y\$12\$5hg4LL0XX9nB2yJUEhIonP9odtCBH2.M9y/...	NULL	2024-07-19 10:59:46	2024-07-19 10:59:46	
7	7	Bobo	bob@bob.com	Pasien	NULL	\$2y\$12\$5hg4LL0XX9nB2yJUEhIonP9odtCBH2.M9y/...	NULL	2024-07-19 10:59:46	2024-07-19 10:59:46	
8	8	raf	rauf@mail.com	Pasien	NULL	\$2y\$12\$5hg4LL0XX9nB2yJUEhIonP9odtCBH2.M9y/...	NULL	2024-07-19 10:59:46	2024-07-19 10:59:46	
9	9	Robert	robert@gmail.com	Pasien	NULL	\$2y\$12\$5hg4LL0XX9nB2yJUEhIonP9odtCBH2.M9y/...	NULL	2024-07-19 10:59:46	2024-07-19 10:59:46	

Gambar 5.21 Penambahan data user setelah registrasi akun pasien dilakukan

Selanjutnya pengujian alternatif. Kasus untuk pengujian alternatif adalah ketika user menginputkan email yang sudah terdaftar pada *website*. User akan diminta menginputkan email lain dikarenakan email tersebut sudah terdaftar. Hasil dari pengujian ini dapat dilihat pada tabel 5.5 dan gambar 5.20

Tabel 5.5 Pengujian alternatif registrasi akun pasien

Kasus dan Hasil Uji (alternatif)	
Data masukan	Mengisikan email yang sudah terdaftar di <i>website</i>
Hasil yang diharapkan	Akan muncul notifikasi yang memberitahu bahwa email yang diinputkan sudah didaftarkan.
Pengamatan	Notifikasi pemberitahuan muncul yang menginformasikan bahwa email sudah didaftarkan
Hasil	Sesuai



Gambar 5.22 Notifikasi saat mendaftarkan akun email yang sudah ada

### 5.2.2.3 Admin Menerima pengajuan reservasi dari pasien

Pada pengujian ini pasien akan mengajukan reservasi melalui akunya. Setelah itu admin akan memeriksa pengajuan ini dan menyetujui/menerima reservasi ini melalui akun admin. Pada kasus ini pasien bernama Robert dan user admin bernama abdul. Hasil dari pengujian ini dapat dilihat pada tabel 5.6 dibawah ini :

Tabel 5.6 Pengujian kondisi benar reservasi dari pasien

<b>Kasus dan Hasil Uji (benar)</b>	
Data masukan	Reservasi dari pasien serta persetujuan dari admin
Hasil yang diharapkan	Data tersimpan dan reservasi yang diajukan akan berubah statusnya menjadi diterima.
Pengamatan	Sistem berhasil menyimpan data dan berhasil mengubah status dari reservasi yang diajukan
Hasil	Sesuai

Pada pengujian ini user pasien perlu mengajukan reservasi terlebih dahulu dengan cara mengisikan form tambah reservasi pada *website*. Setelah selesai diinputkan maka yang data akan muncul pada reservasi dengan status belum diproses dan muncul pada draft reservasi dari pasien. Setelah itu admin akan menerima pengajuan yang membuat status reservasi pada pasien akan berubah menjadi diterima dan data reservasi pada admin akan dipindahkan ke bagian reservasi dari admin. Proses ini dapat dilihat pada gambar 5.23, 5.24, 5.25, 5.26.

Gambar 5.23 Tampilan form saat pasien melakukan reservasi

No	Nama Pasien	Lokasi Perawatan	Nama Dokter	Tanggal	Jam	Status
1	robert	Padang	Alif	2024-07-22	10:00:00	Belum diproses

Gambar 5.24 Tampilan history pasien selesai melakukan reservasi

No	Nama Pasien	Lokasi Perawatan	Nama Dokter	Tanggal	Jam	Aksi
1	anto	Padang	Alif	2024-07-09	18:00:00	<button>Terima</button> <button>Tolak</button>
2	robert	Padang	Alif	2024-07-22	10:00:00	<button>Terima</button> <button>Tolak</button>

Showing 1 to 2 of 2 entries

Copyright © Xenon Dental House

Gambar 5.25 Tampilan user admin setelah pasien selesai melakukan reservasi

No	Nama Pasien	Lokasi Perawatan	Nama Dokter	Tanggal	Nomor HP	Jam	Rekam Medik	Aksi
1	robert	Padang	Alif	2024-07-22	62822880101	10:00:00		<button>Edit</button> <button>Hapus</button>
2	Anisa	Padang	Dr. Andi Wijayaa	2024-07-18	62822050022	11:00:00		<button>Edit</button> <button>Hapus</button>
3	Andy	Padang	Dr. Andi Wijayaa	2024-07-18	62822881222	13:00:00		<button>Edit</button> <button>Hapus</button>
4	Siti Aminah	Padang	Alif	2024-07-15	6289617702747	11:30:00		<button>Edit</button> <button>Hapus</button>
5	Budi	Padang	Alif	2024-07-09	628654345676543	17:00:00		<button>Edit</button> <button>Hapus</button>
6	Badu	Padang	Alif	2024-07-01	628545215621	10:00:00		<button>Edit</button> <button>Hapus</button>

Showing 1 to 6 of 6 entries

[Tambahkan Reservasi](#)

Copyright © Xenon Dental House

Gambar 5.26 Tampilan data reservasi admin setelah reservasi pasien diterima

No	Nama Pasien	Lokasi Perawatan	Nama Dokter	Tanggal	Jam	Status
1	robert	Padang	Alif	2024-07-22	10:00:00	Diterima

Showing 1 to 1 of 1 entries

Copyright © Xenon Dental House

Gambar 5.27 Tampilan *history* reservasi pasien setelah reservasi pasien diterima

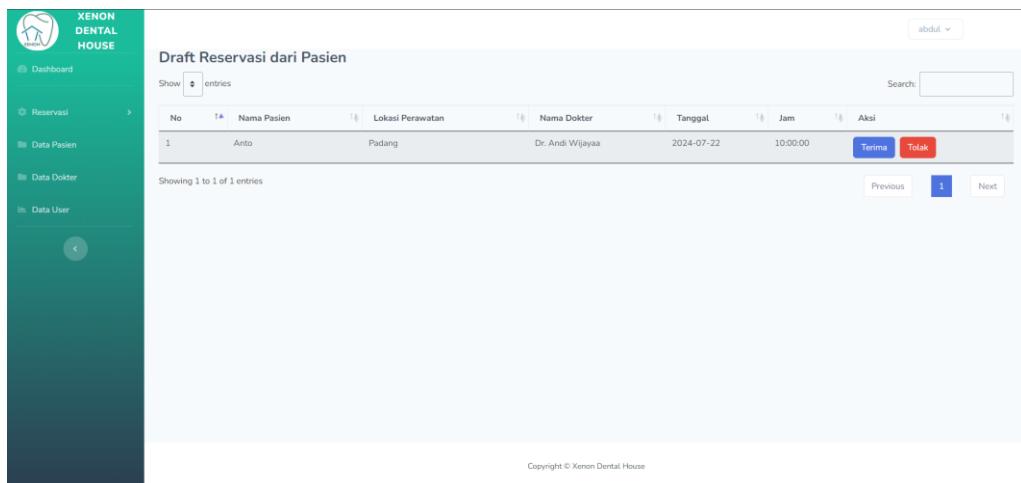
Selanjutnya pengujian alternatif. Pada pengujian ini admin juga bisa menolak pengajuan dari pasien dengan cara menekan tombol tolak. Status reservasi pada akun pasien akan menjadi ditolak serta data akan hilang dari tampilan admin. Hasil dari pengujian ini dapat dilihat pada tabel 5.7 dan gambar 5.26 , 5.27, 5.28.

Tabel 5.7 Pengujian alternatif reservasi dari pasien

<b>Kasus dan Hasil Uji (alternatif)</b>	
Data masukan	Dokter menolak pengajuan reservasi dari pasien
Hasil yang diharapkan	Status reservasi dari pasien akan berubah menjadi ditolak serta data akan hilang dari tampilan admin
Pengamatan	Status reservasi pada pasien adalah ditolak dan data sudah tidak ada pada tampilan admin
Hasil	Sesuai

No	Nama Pasien	Lokasi Perawatan	Nama Dokter	Tanggal	Jam	Status
1	Anto	Padang	Dr. Andi Wijayaa	2024-07-22	10:00:00	Belum diproses

Gambar 5.28 Tampilan history reservasi pasien setelah melakukan reservasi



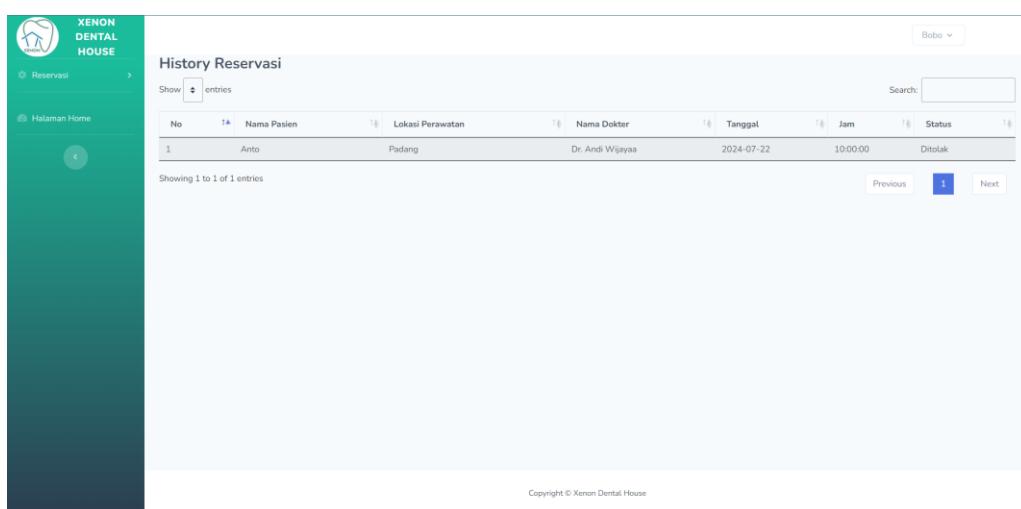
The screenshot shows a table titled "Draft Reservasi dari Pasien". The columns are: No, Nama Pasien, Lokasi Perawatan, Nama Dokter, Tanggal, Jam, and Aksi. There is one entry: No 1, Nama Pasien: Anto, Lokasi Perawatan: Padang, Nama Dokter: Dr. Andi Wijayaa, Tanggal: 2024-07-22, Jam: 10:00:00. The "Aksi" column contains two buttons: "Terima" (blue) and "Tolak" (red). The footer says "Copyright © Xenon Dental House".

No	Nama Pasien	Lokasi Perawatan	Nama Dokter	Tanggal	Jam	Aksi
1	Anto	Padang	Dr. Andi Wijayaa	2024-07-22	10:00:00	<button>Terima</button> <button>Tolak</button>

Showing 1 to 1 of 1 entries

Copyright © Xenon Dental House

Gambar 5.29 Tampilan reservasi pasien setelah melakukan reservasi pada admin



The screenshot shows a table titled "History Reservasi". The columns are: No, Nama Pasien, Lokasi Perawatan, Nama Dokter, Tanggal, Jam, Status, and Aksi. There is one entry: No 1, Nama Pasien: Anto, Lokasi Perawatan: Padang, Nama Dokter: Dr. Andi Wijayaa, Tanggal: 2024-07-22, Jam: 10:00:00, Status: Ditolak. The footer says "Copyright © Xenon Dental House".

No	Nama Pasien	Lokasi Perawatan	Nama Dokter	Tanggal	Jam	Status	Aksi
1	Anto	Padang	Dr. Andi Wijayaa	2024-07-22	10:00:00	Ditolak	

Showing 1 to 1 of 1 entries

Copyright © Xenon Dental House

Gambar 5.30 Tampilan histori reservasi pasien setelah ditolak admin

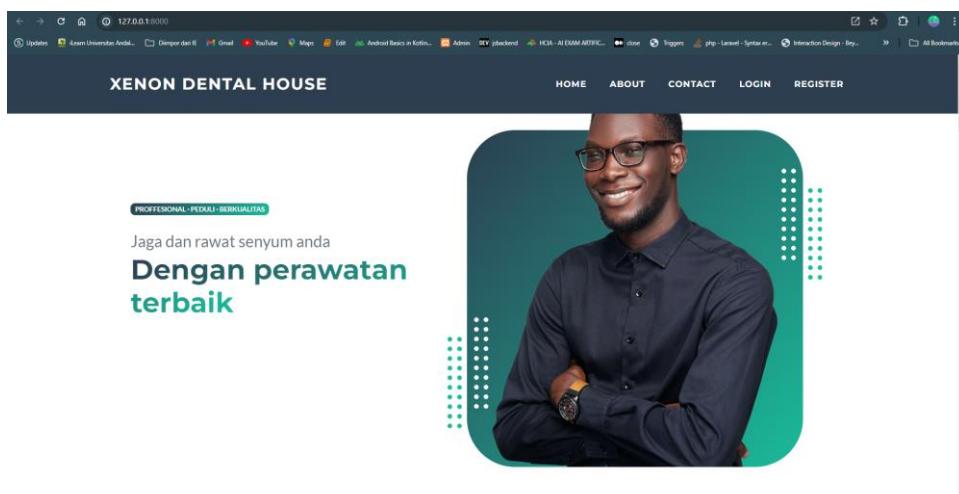
#### 5.2.2.4 Pengujian mode offline PWA pada website

Pada pengujian ini dilakukan uji coba mode offline dari *website* yang menggunakan pendekatan PWA(Progressive Web Apps). Pengujian ini dilakukan ketika kondisi server mati atau user tidak bisa mengakses server dari *website*.

Tabel 5.8 Pengujian mode offline pada *website*

<b>Kasus dan Hasil Uji (benar)</b>	
Kondisi pengujian	Server dari <i>website</i> akan dimatikan
Hasil yang diharapkan	<i>Website</i> tetap bisa menampilkan landing page dari <i>website</i> dengan cara menyimpan cache dari <i>website</i> .
Pengamatan	Sistem berhasil menampilkan <i>website</i> dalam kondisi offline walaupun setelah dimuat ulang.
Hasil	Sesuai

Pada pengujian ini pertama user menampilkan kondisi ketika *website* ketika server masih aktif. Setelah itu dilakukan uji coba dengan mematikan server Laravel serta mengaktifkan fitur offline pada chrome. Setelah itu *website* akan dimuat ulang dalam kondisi server tidak aktif. Hasilnya *website* akan tetap bisa menampilkan landing page dari *website* dengan fungsional yang terbatas.



Gambar 5.31 Tampilan landing page saat server aktif

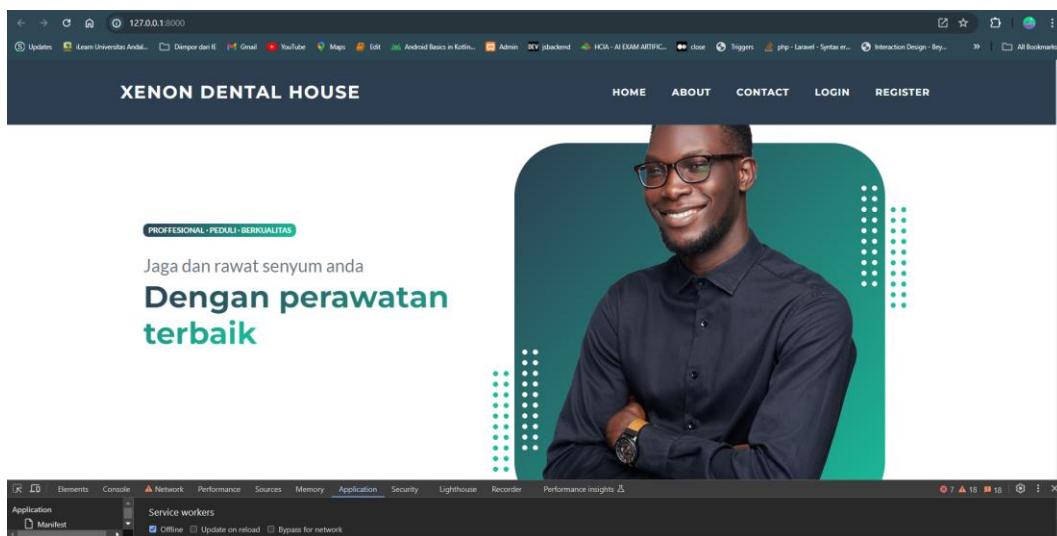
```

1 > const CACHE_NAME = 'pwa-cache-v1';
2 > const urlToCache = [
3   '/',
4   '/css/sb-admin-2.min.css',
5   '/css/styles.css',
6   '/css/stylesdokter.css',
7   '/css/stylesfoto.css',
8   '/js/scripts.js',
9   '/js/scriptsfoto.js',
10  '/img/1.jpg',
11  '/img/2.jpg',
12  '/img/3.jpg',
13  '/img/4.jpg',
14  '/img/testimonials-1.jpg',
15  '/img/testimonials-2.jpg',
16  '/img/testimonials-3.jpg',
17  '/img/portfolio/bleaching.jpg',
18  '/img/portfolio/bracesultra.jpg',
19  '/img/portfolio/denture.png'

```

Server running on [http://127.0.0.1:8000].  
Press Ctrl+C to stop the server

Gambar 5.32 Tampilan saat server dimatikan



Gambar 5.33 Tampilan landing page tetap muncul setelah server mati

### 5.2.3 Kesimpulan hasil pengujian

Dari hasil pengujian aplikasi web yang telah dilakukan menunjukkan bahwa *website* yang dikembangkan telah sesuai dengan rancangan sistem yang direncanakan. Pengujian menggunakan metode black box mengonfirmasi bahwa semua fungsionalitas beroperasi sesuai harapan, dengan input dan output yang tepat pada setiap item yang diuji. Oleh sebab itu, dapat disimpulkan bahwa aplikasi sistem informasi manajemen klinik gigi Xenon Dental House telah berjalan sesuai dengan kebutuhan fungsional yang telah dirancang. Secara keseluruhan, aplikasi

web memenuhi semua kriteria pengujian yang telah ditetapkan, menunjukkan bahwa aplikasi siap untuk digunakan. Kesimpulan hasil pengujian dapat dilihat pada tabel 5.9 berikut.

Tabel 5.9 Kesimpulan hasil pengujian

No	Item Uji	Jenis Aplikasi	Pengguna	Detail Pengujian	Hasil
1	Authentikasi dan Pengelolaan Akun	Website	Semua User	Login, ganti password, logout, ganti informasi profil, menghapus akun pribadi	Sesuai
2	Lihat Dasboard	Website	Admin, Owner, Dokter	Lihat, Filter	Sesuai
3	Konfirmasi reservasi dari Pasien	Website	Admin	Lihat, Tolak, terima	Sesuai
4	Reservasi pasien	Website	Admin	Lihat, tambah, edit, hapus	Sesuai
5	Mengelola data pasien	Website	Admin	Lihat, edit, hapus	Sesuai
6	Mengelola data rekam medik	Website	Admin	Lihat, edit	Sesuai
7	Mengelola data dokter, jadwal dokter	Website	Admin	Lihat, tambah, edit, hapus	Sesuai
8	Mengelola data user	Website	Admin	Lihat, tambah, edit, hapus	Sesuai
9	Registrasi Akun	Website	Pasien	Registrasi akun baru	Sesuai
10	Reservasi klinik	Website	Pasien	Tambah, Edit	Sesuai
11	Melihat Riwayat reservasi	Website	Pasien	Lihat	Sesuai
12	Lihat Jadwal Praktek	Website	Dokter	Lihat	Sesuai
13	Mengelola data pasien	Website	Dokter	Lihat, edit, hapus	Sesuai

Tabel 5.9 Kesimpulan hasil pengujian (lanjutan)

14	Mengajukan permintaan izin	<i>Website</i>	Dokter	Lihat, tambah, edit, hapus	Sesuai
15	Melihat data profit klinik	<i>Website</i>	Owner	Lihat, Filter	Sesuai
16	Melihat jumlah kunjungan pasien	<i>Website</i>	Owner	Lihat, Filter	Sesuai
17	Melihat jumlah penanganan perawatan berdasarkan kategori	<i>Website</i>	Owner	Lihat, Filter	Sesuai
18	Melihat performa dokter berdasarkan jumlah pasien	<i>Website</i>	Owner	Lihat, Filter	Sesuai
19	Melihat statistik pasien berdasarkan umur	<i>Website</i>	Owner	Lihat, Filter	Sesuai
20	Melihat Data dokter	<i>Website</i>	Owner	Lihat	Sesuai
21	Melihat Data Pasien	<i>Website</i>	Owner	Lihat	Sesuai
22	Melihat data Pemasukan dan keuangan klinik	<i>Website</i>	Owner	Lihat	Sesuai
23	Memutuskan perizinan dokter	<i>Website</i>	Owner	Terima, Tolak	Sesuai
24	Mengelola pencatatan pengeluaran klinik	<i>Website</i>	Owner	Lihat, tambah, edit, hapus	Sesuai
25	Mengelola harga perawatan	<i>Website</i>	Owner	Lihat, tambah, edit, hapus	Sesuai

#### **5.2.4 Analisa dan Pembahasan Hasil**

Sistem informasi yang dikembangkan memiliki beberapa fitur yang dirancang untuk mendukung pengelolaan klinik dan layanan kesehatan dengan lebih efektif dibandingkan sistem yang ada sebelumnya. Berikut analisis perbandingan antara sistem manual yang sebelumnya digunakan dengan sistem berbasis web yang telah dikembangkan.

##### **1. Sentralisasi data dan Terorganisir**

Sistem sebelumnya menggunakan berbagai media untuk mencatat data, mulai dari catatan manual hingga reservasi melalui WhatsApp. Tidak adanya satu pusat data yang terintegrasi dapat diatasi dengan dibangunnya website ini dalam mengakses dan memperbarui informasi. Sistem informasi manajemen berbasis web ini memusatkan semua data pasien, dokter, layanan, serta rekam medis dalam satu platform digital yang dapat diakses oleh pihak yang berwenang serta akses *realtime* memungkinkan klinik mengakses data antar cabang pada klinik tanpa memandang lokasi.

##### **2. Keamanan dan Akurasi Data**

Pada sistem sebelumnya, keamanan data menjadi perhatian penting karena saat ini data seperti rekam medik masih disimpan dalam bentuk fisik(kertas). Data yang tersimpan dalam bentuk fisik (kertas) atau melalui WhatsApp tidak memiliki kontrol akses yang ketat, sehingga rawan terhadap akses tidak sah atau bahkan kehilangan. Dengan penerapan sistem informasi ini, keamanan informasi lebih terjamin melalui pengaturan hak akses yang terkelola dan enkripsi data. Selain itu, data yang diinput ke dalam sistem memiliki mekanisme validasi yang meminimalkan kesalahan input, sehingga kualitas dan akurasi data lebih terjaga.

##### **3. Otomasi Penjadwalan dan Reservasi**

Sebelumnya, reservasi pasien dilakukan secara manual melalui WhatsApp, yang dapat menyebabkan miskomunikasi atau data duplikat. Sistem berbasis web

ini mengotomatisasi proses reservasi, memastikan bahwa jadwal dokter selalu terupdate dan tidak terjadi kesalahan dalam penjadwalan. Pasien dapat langsung melihat ketersediaan dokter dan memilih waktu yang sesuai serta membuat sistem lebih transparan.

#### 4. Fitur Visualisasi Data

Sistem sebelumnya tidak memiliki sistem untuk memantau performa klinik secara terukur, sehingga dapat menghambat *owner* untuk melakukan pemantauan berbasis data terkait produktivitas dokter, tren layanan perawatan, dan tren pasien. Fitur visualisasi data dalam sistem berbasis web ini memungkinkan klinik menghasilkan visualisasi yang dapat diakses kapan saja untuk memantau perkembangan klinik secara berkala. Data yang dihasilkan dapat digunakan untuk mendukung keputusan strategis owner.

#### 5. Pengurangan Penggunaan Kertas dan Penyimpanan Fisik

Pada sistem sebelumnya, pengelolaan dokumen secara fisik memerlukan banyak ruang penyimpanan dan kebutuhan akan kertas. Selain itu, risiko kerusakan atau kehilangan dokumen juga tinggi. Sistem ini mengurangi kebutuhan akan penyimpanan fisik, memungkinkan klinik menyimpan data secara digital dan aman, sekaligus mendukung keberlanjutan lingkungan dengan mengurangi penggunaan kertas.

Selain itu, pada sistem ini juga terdapat keunggulan atau kebaruan yang diharapkan dapat memberikan nilai tambah pada tugas akhir ini. Keunggulan dan kebaruan dari sistem ini dapat dilihat sebagai berikut:

##### 1. PWA untuk Akses *Offline*

Sistem ini menggunakan Progressive Web App (PWA) yang memungkinkan halaman web tetap diakses meskipun koneksi internet tidak tersedia. Fitur ini meningkatkan aksesibilitas dan memberikan pengalaman seperti aplikasi mobile. Implementasi ini menggunakan *service workers* untuk mengontrol *caching* dan sinkronisasi data, memungkinkan aplikasi tetap berfungsi secara offline. Pada fitur ini website awalnya akan mencoba mengakses tampilan website melalui server terlebih dahulu, jika website terkoneksi maka akan tampilan akan

ditampilkan berdasarkan server. Ketika website tidak terhubung ke server maka website akan menampilkan tampilan dari *cache* yang sudah disimpan sebelumnya, jika cache juga tidak ada maka website akan menampilkan tampilan *default* website offline yang sudah deprogram sebelumnya. Dengan adanya fitur ini maka pengguna dapat mengakses informasi penting tanpa bergantung pada koneksi internet yang stabil.

## 2. Notifikasi WhatsApp (API Twilio)

Sistem ini mengintegrasikan API Twilio untuk memfasilitasi pengiriman notifikasi langsung melalui platform WhatsApp. Integrasi ini bertujuan untuk membantu komunikasi antara sistem dan pengguna. Dengan fitur ini maka klinik bisa mengirim pesan whatsapp secara terprogram. Penggunaan API Twilio juga bisa mengirimkan *template* pesan yang dipersonalisasi sesuai kebutuhan dengan berbagai jenis konten termasuk teks, gambar, video dan *file* serta sistem dapat menangani pengiriman pesan dalam jumlah besar. Dengan adanya fitur ini pasien mendapatkan pemberitahuan terkait jadwal dan reservasi secara real-time, meningkatkan komunikasi antara klinik dan pasien.

## 3. Visualisasi Data dengan Filter

Fitur visualisasi data memudahkan pemilik dan manajemen klinik untuk melihat statistik klinik, seperti statistic pasien, pendapatan bulanan, jumlah kunjungan, kinerja dokter. Data ini dapat difilter berdasarkan bulan untuk analisis kinerja yang lebih detail.

## 4. Laravel Versi 11

Sistem dibangun menggunakan Laravel 11, yang memberikan peningkatan dalam hal kesederhanaan struktur aplikasi, kinerja, keamanan, kompatibilitas dengan PHP terbaru, kemudahan pengujian, dan konfigurasi.

Disamping keunggulan tersebut, juga terdapat kekurangan pada sistem informasi manajemen klinik yang sudah dibangun ini dan dapat dilakukan pengembangan lebih lanjut menjadi lebih baik. Kekurangan dalam pembangunan sistem ini terletak pada belum tersedianya fitur payment gateway yang memungkinkan pasien melakukan transaksi secara online. Hal ini menjadi penting

untuk membantu proses pembayaran dan memberikan kenyamanan lebih bagi pengguna. Selain itu, meskipun sistem telah menggunakan teknologi PWA, penerapannya masih terbatas hanya pada fitur offline mode dan diperlukan peningkatan keamanan untuk fitur offline ini. Fitur penting lainnya dari PWA seperti push notification, background sync, dan peningkatan user experience lainnya juga belum diimplementasikan.

## **BAB VI**

### **KESIMPULAN DAN SARAN**

Bab ini menguraikan kesimpulan dan saran dari laporan tugas akhir ini. Kesimpulan adalah hasil yang dicapai dari penelitian berdasarkan tujuan yang telah ditetapkan di awal penelitian. Saran adalah rekomendasi yang diharapkan terkait penelitian yang telah dilakukan untuk masa mendatang.

#### **6.1 Kesimpulan**

*Website* sistem informasi manajemen klinik gigi pada klinik Xenon Dental House telah berhasil dibangun menggunakan metode waterfall. Dari penelitian ini, berhasil dibangun sebuah sistem informasi manajemen klinik gigi di Klinik Gigi Xenon Dental House dalam mengatasi kendala serta membantu pengelolaan manajemen klinik. Sistem ini mencakup pengelolaan data pasien, dokter, data rekam medis, reservasi, izin dokter, layanan perawatan, dan visualisasi data klinik. Dengan implementasi sistem ini, efisiensi operasional klinik meningkat, memastikan semua aspek manajemen dapat dikelola dengan lebih baik, lebih aman dan terstruktur.

#### **6.2 Saran**

Sistem informasi manajemen klinik gigi pada klinik Xenon Dental House ini masih membutuhkan pengembangan lebih lanjut pada fungsional aplikasi. Saat ini *website* yang dibangun hanya fokus pada penerapan *website* pada bisnis proses klinik gigi xenon dental house. Saran terhadap pembangunan *website* ini selanjutnya adalah pembangunan fitur payment gateway pada *website* ini agar bisa dilakukan transaksi secara online. Pembangunan lebih lanjut juga dengan menerapkan fitur PWA(Progressive Web App) lainnya seperti push notification, background sync dan fitur lainnya yang ada pada PWA karena pada pembangunan *website* ini hanya berfokus pada fitur offline PWA saja serta meningkatkan keamanan website untuk fitur PWA offline.

## **DAFTAR PUSTAKA**

- 'Afiifah, K., Azzahra, Z.F. and Anggoro, A.D. (2022) 'Analisis Teknik Entity-Relationship Diagram dalam Perancangan Database Sebuah Literature Review', Intech, 3(2), pp. 18–22. doi:10.54895/intech.v3i2.1682.
- Aleksandrs Hodakovskis; Katrina Antonova (2019) 'The ultimate guide to Progressive Web Apps', Scandiweb.
- Aprilli; Inayati, Aini; Fauzi, Feddi Dea. (2018). Desain Database Sistem Informasi Rekam Medis Berbasis Microsoft Access Di Klinik Dokter Gigi. Jurnal Hospita Science 2(2) 1-8
- Astuti, R. (2009) 'Pemodelan Analisis Berorientasi Objek dengan Use Case', Media Informatika, 8(2), pp. 73–81.
- Dharwiyanti, S. (2003) 'Pengantar Unified Modeling Language (UML)', pp. 1–13.
- Hariyanto, S. (2018) 'Sistem Informasi Manajemen', Sistem Informasi Manajemen, 9(1), pp. 80–85.
- Haryanto, D. and Saputra Elsi, Z.R. (2021) 'Analisis Performance Progressive Web Apps Pada Aplikasi Shopee', Jurnal Ilmiah Informatika Global, 12(2), pp. 106–111. doi:10.36982/jiig.v12i2.1944.
- Ismanto, I., Hidayah, F. and Charisma, K. (2020) 'Pemodelan Proses Bisnis Menggunakan Business Process Modelling Notation (BPMN) (Studi Kasus Unit Penelitian Dan Pengabdian Kepada Masyarakat (P2KM) Akademi Komunitas Negeri Putra Sang Fajar Blitar)',
- Karli, T., Muawwal, A. and Thayf, M.S.S. (2023) 'Implementasi Progressive Web Application Pada Website Frizfoo Menggunakan Express.Js', KHARISMA Tech, 18(2), pp. 110–124. doi:10.55645/kharismatech.v18i2.445.
- Kemenkes RI 2011 (2011) 'Permenkes No. 028 tentang Klinik 2011', Menteri Kesehatan Republik Indonesia Peraturan Menteri Kesehatan Republik Indonesia, Nomor 65(879), pp. 2004–2006.

- Mahdalena, M., Alamsyah, N. and Sidik, A. (2023) ‘Sistem Informasi Manajemen Dan Keuangan Berbasis Web Pada Klinik Gigi Eldental Banjarmasin’, Jurnal Sains Sistem Informasi, 1(1), p. 25. doi:10.31602/jssi.v1i1.9672.
- Majidah, R., Rusdianto, D.S. and ... (2019) ‘Pengembangan Sistem Informasi Pengelolaan Klinik Gigi Berbasis Website Menggunakan Prinsip Point of Sale
- Media Informatika, 8(2), pp. 73–81. Available at:  
[https://jurnal.liksi.ac.id/Jurnal/7\\_2009/Pemodelan\\_Analisis\\_rini\\_.pdf](https://jurnal.liksi.ac.id/Jurnal/7_2009/Pemodelan_Analisis_rini_.pdf).
- Muddin, S., Tehuayo, H. and Iksan, F. (2021) ‘Penerapan Teknologi Progressive Web Apps (PWA) Pada Sistem Informasi Sma Negeri 7 Buru Selatan’, Jurnal Teknologi dan Komputer (JTEK), 1(01), pp. 16–23.  
doi:10.56923/jtek.v1i01.48.
- Muntihana, V. et al. (2017) Berbasis Web Dan Android Pada Klinik Gigi Lisda.
- Novianti Indah. 2021. Peran Teknologi Informasi Pada Perubahan Organisasi Dan Fungsi Akuntansi Manajemen. 1
- Nuryani, S. (2021) ‘Pengembangan Aplikasi Mobile Booking Online Perawatan Gigi Dengan Metode Prototype Studi Kasus Di Klinik Gigi Budiono, Drg. Kota Bandung’, Jurnal Ekonomi, Sosial & Humaniora, 2(06), pp. 18–28.
- Pricillia, T. and Zulfachmi (2021) ‘Perbandingan Metode Pengembangan Perangkat Lunak (Waterfall, Prototype, RAD)’, Jurnal Bangkit Indonesia, 10(1), pp. 6–12. doi:10.52771/bangkitindonesia.v10i1.153. 19
- Pulungan, S.M. et al. (2023) ‘Analisis Teknik Entity-Relationship Diagram Dalam Perancangan Database’, Jurnal Ekonomi Manajemen dan Bisnis (JEMB), 1(2), pp. 98–102. doi:10.47233/jemb.v1i2.533.
- Rendra Wisiasto 2010. Perancangan Dan Implementasi Sistem Penjadwalan Konsultasi Dokter Terhadap Pasien Via Sms Gateway Di Rumah Sakit Rajawali. Universitas Komputer Indonesia.
- Riandy, G., Edi, S., & Rengga, A. (2009). Rancang Bangun Sistem Informasi

Reservasi Online Fisioterapi Menggunakan JSP Sebagai Sistem Pelayanan Terpadu. EEPIS Final Project.

Sari, Ira Puspita & Diki Arisandi.(2017). Sistem Informasi Manajemen Klinik Gigi Berbasis Client Server.

Setyadi, H.A. and Perbawa, D.S. (2021) ‘Sistem Informasi Rekam Medis Di Klinik Gigi Rumah Sakit Paru dr. Ario Wirawan Salatiga’, Jurnal Infortech, 3(2), pp. 110–116. doi:10.31294/infortech.v3i2.11209.

Silberschatz, A., Korth, H.F. and Sudarshan, S., 2011. *Database system concepts*. 6th ed. New York: McGraw-Hill.

Sommerville, I. (2011) Requisitos de ingeniería de software.

Stiehl, V., Raw, R. and Smith, P. (2014) Process-driven applications with BPMN, Process-Driven Applications with BPMN. doi:10.1007/978-3-319-07218-0.

Tjiptono, Fandy. 2007. Strategi Pemasaran. Yogyakarta: Edisi II. Andi.

Wardianto, M. (2011). Rancang bangun aplikasi pendapatan online jasa pengobatan berbasis multimedia pada Klinik Utama Siti Aksar Depok.

Wahyudi, T., Supriyanta, S. and Faqih, H. (2021) ‘Pengembangan Sistem Informasi Presensi Menggunakan Metode Waterfall’, Indonesian Journal on Software Engineering (IJSE), 7(2), pp. 120–129.

Wagner, G. (2017) ‘Information and Process Modeling for Simulation – Part I: Objects and Events’, Journal of Simulation Engineering, 1(1), pp. 1–25. Available at: <https://articles.jsime.org/1/1>.

Warrender, R., Links, U. and Dutton, S. (2018). ‘Progressive Web Apps: the future of the mobile web’.

Wijoyo, H. (2021) Sistem Informasi Manajemen, Buku.

Yohana, N.D. and Marisa, F. (2018) ‘Perancangan Proses Bisnis Sistem Human Resource Management (HRM) Untuk Meningkatkan Kinerja Pegawai’, J I M P - Jurnal Informatika Merdeka Pasuruan, 3(2), pp. 23–32. doi:10.37438/jimp.v3i2.168.

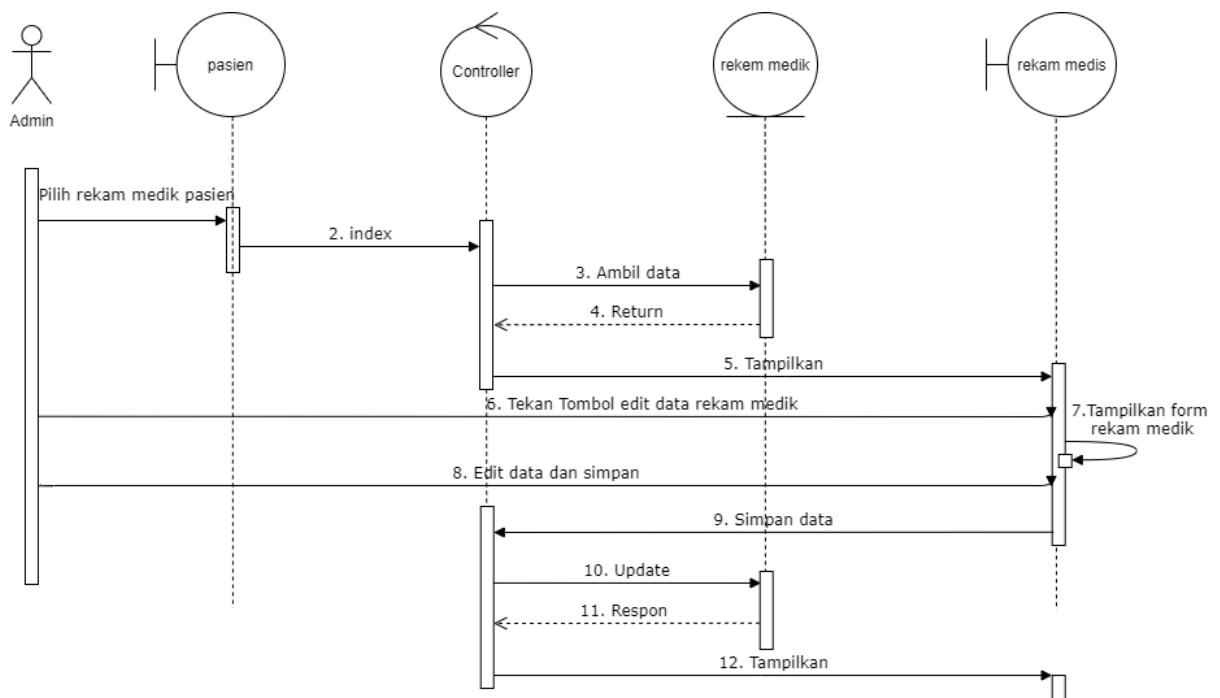
**LAMPIRAN A**

***SEQUENCE DIAGRAM***

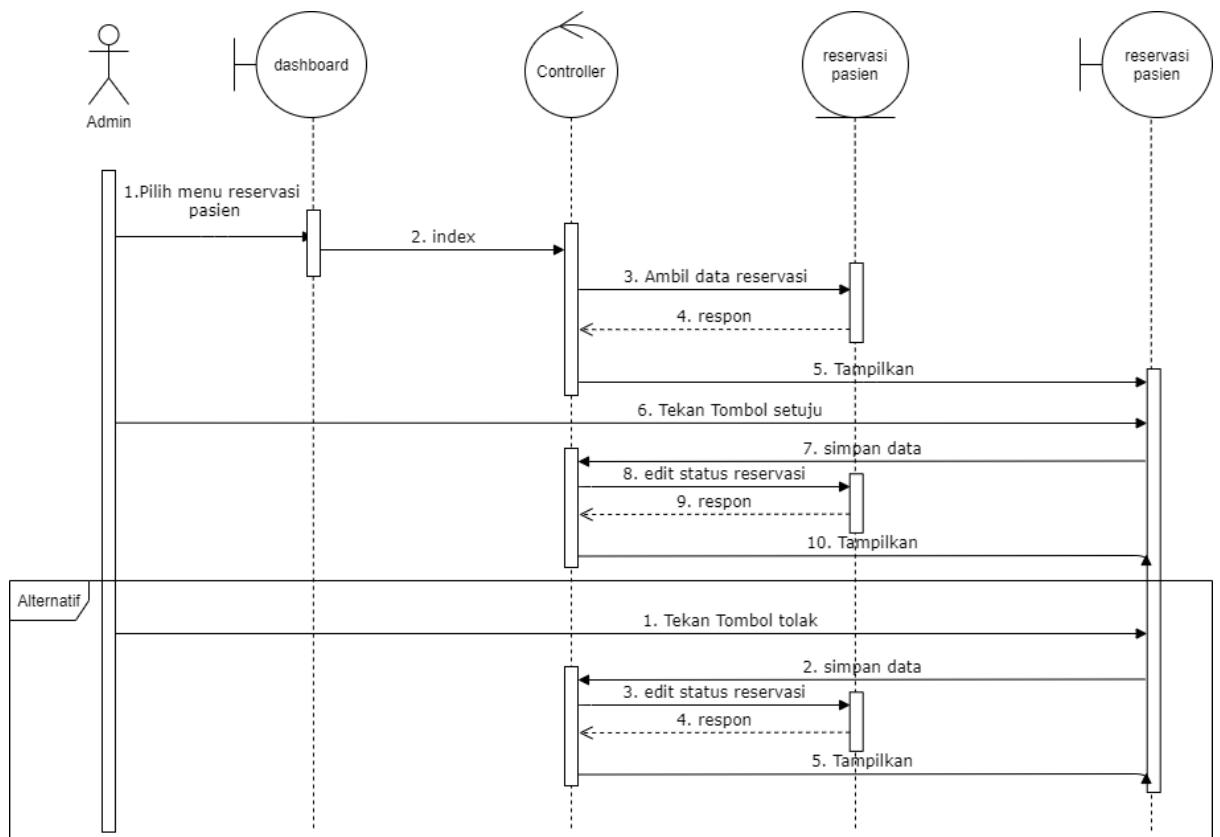
**(LANJUTAN)**

## Admin

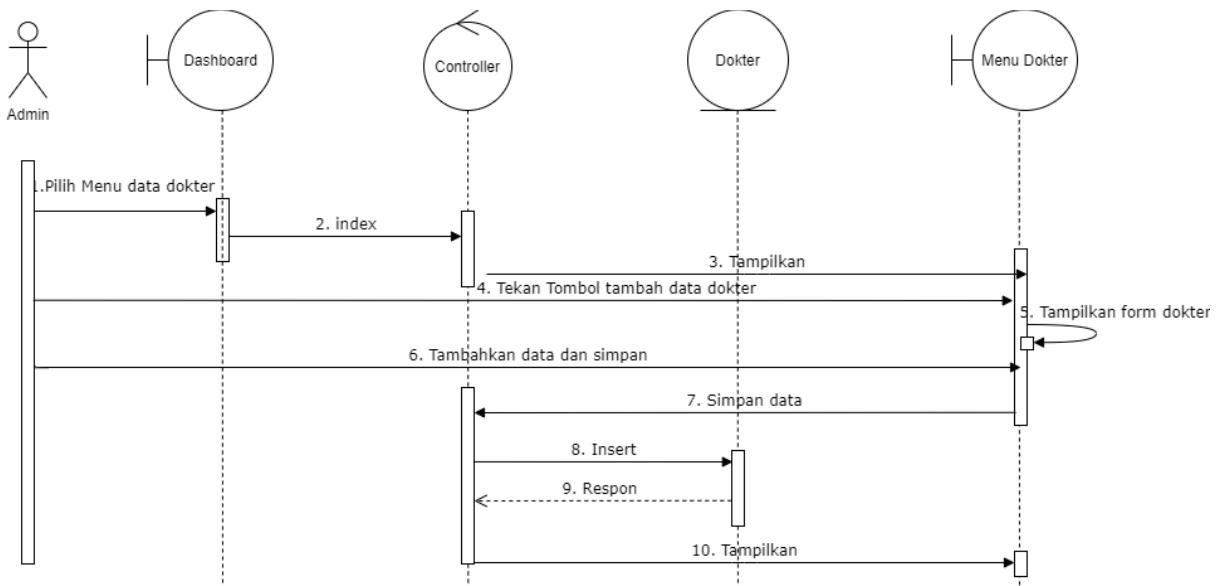
### 1. Edit rekam medik



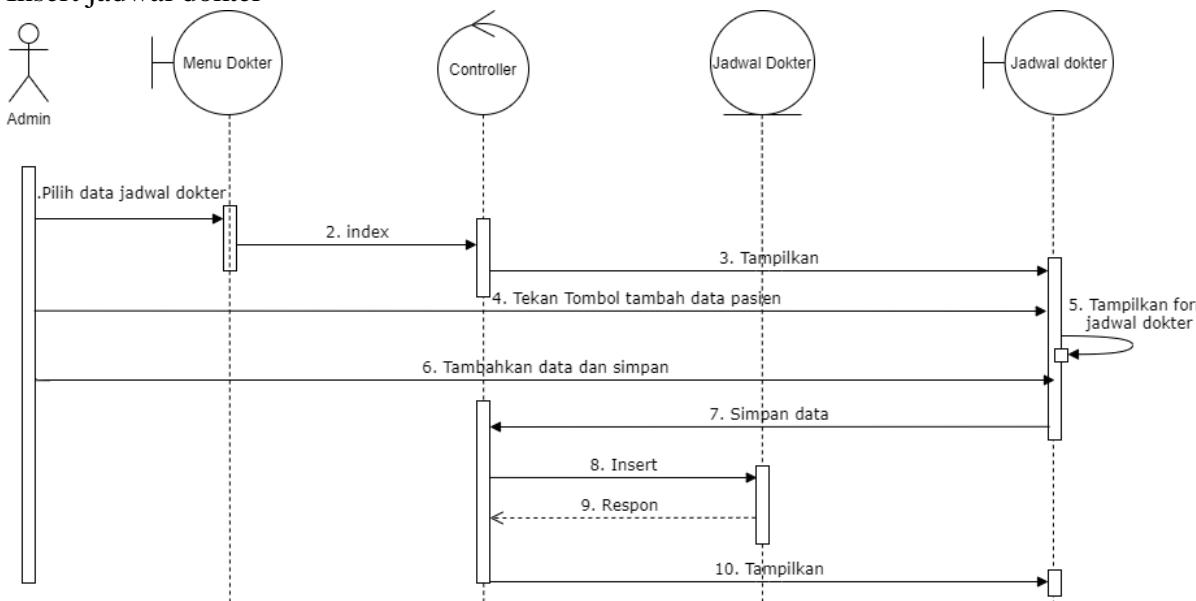
### 2. Menyetujui reservasi



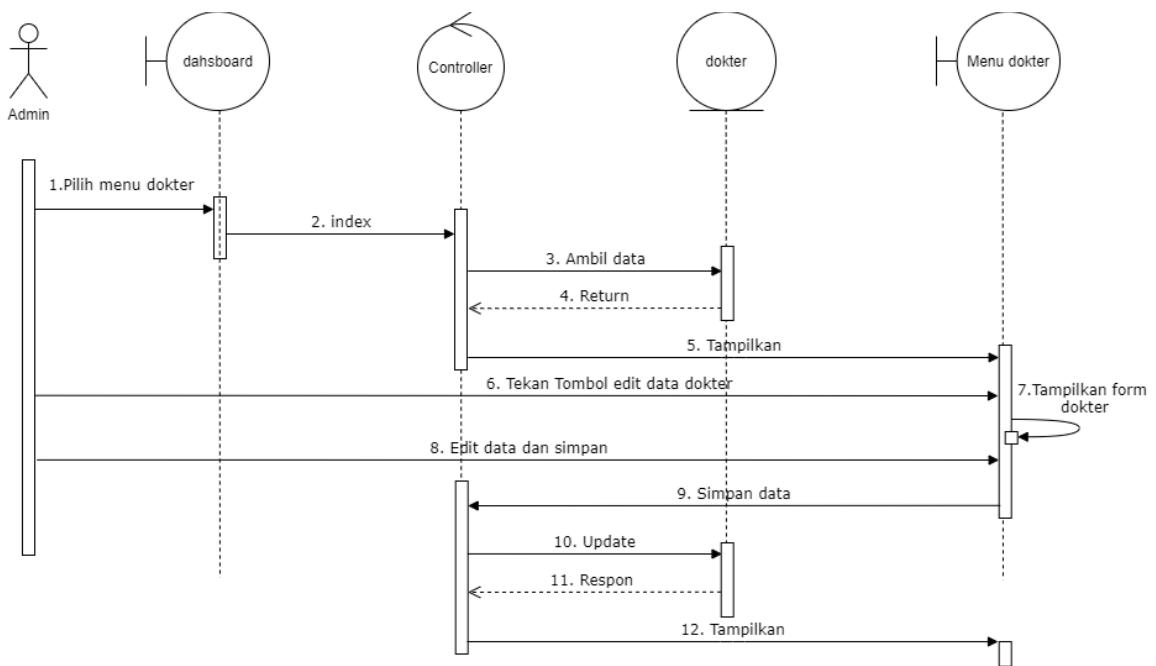
### 3. Insert data dokter



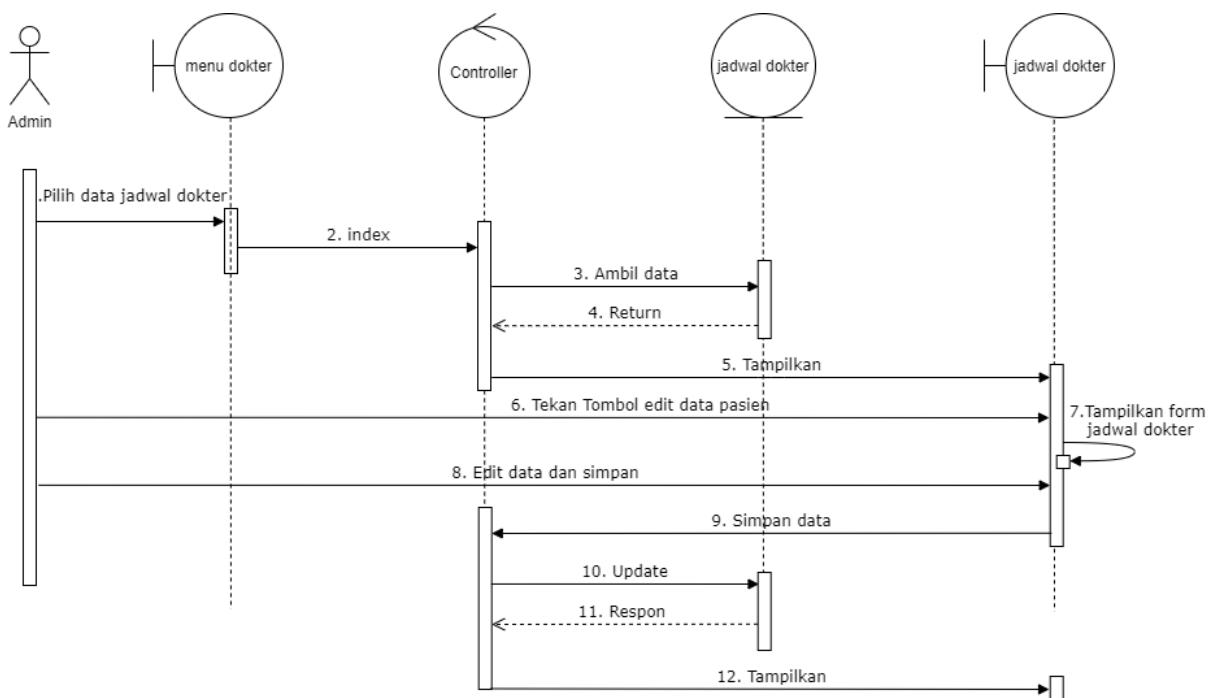
### 4. Insert jadwal dokter



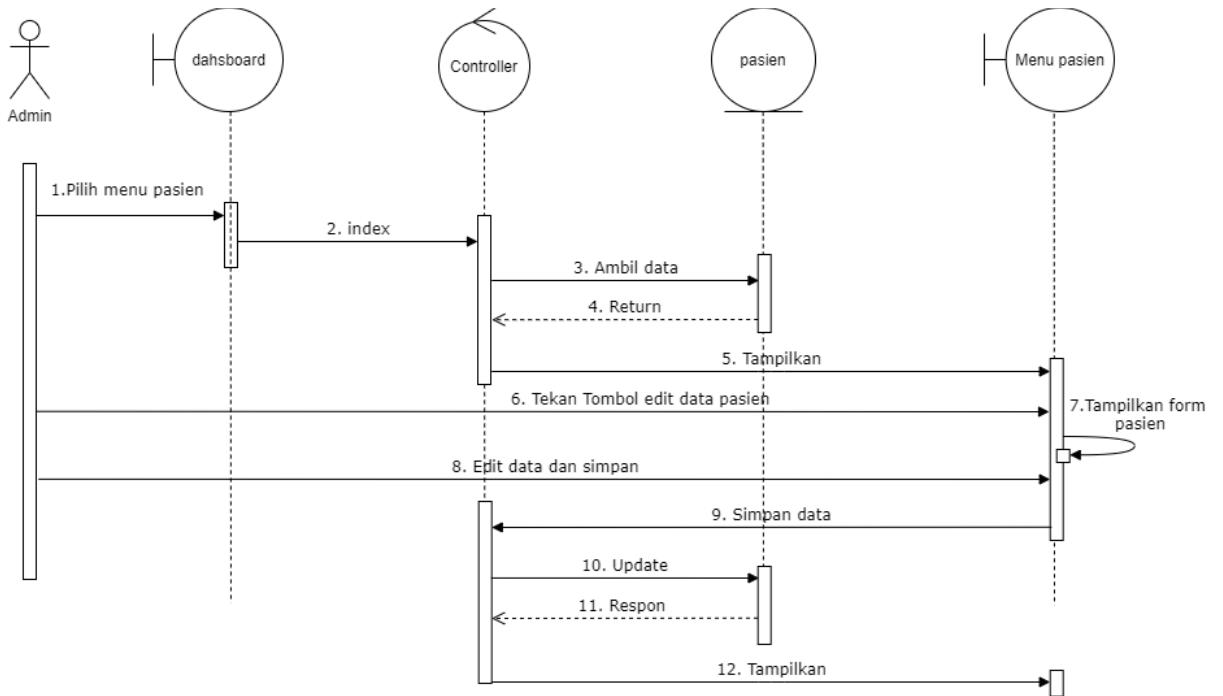
## 5. Edit data dokter



## 6. Edit dokter jadwal

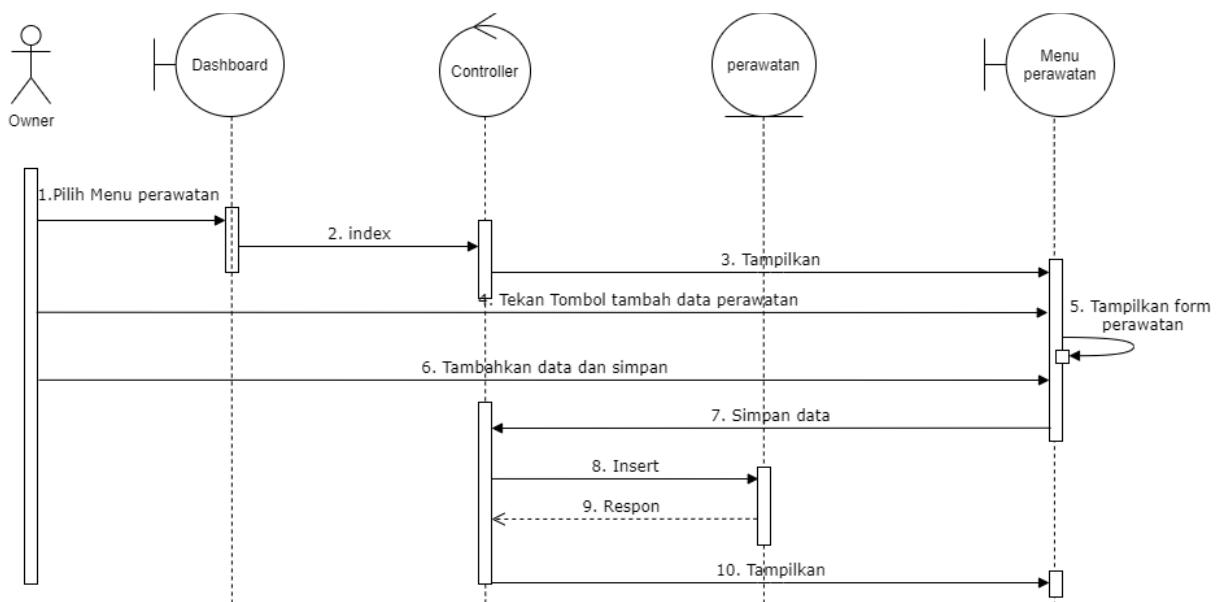


## 7. Update data pasien

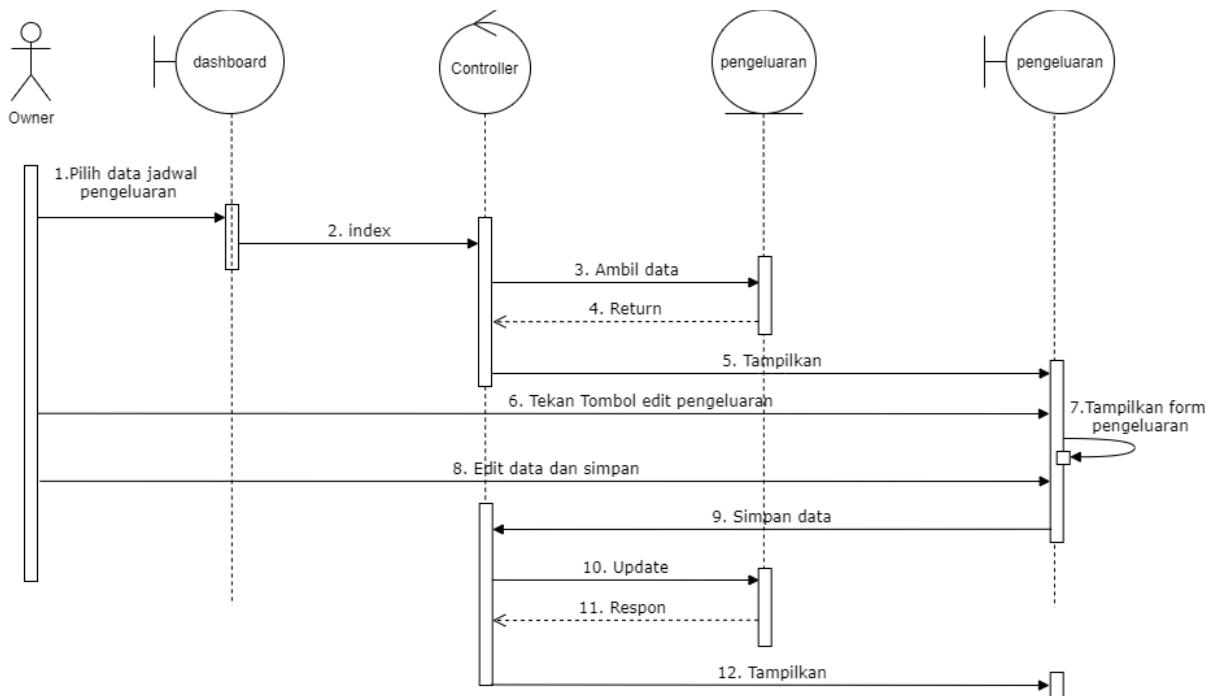


User Owner

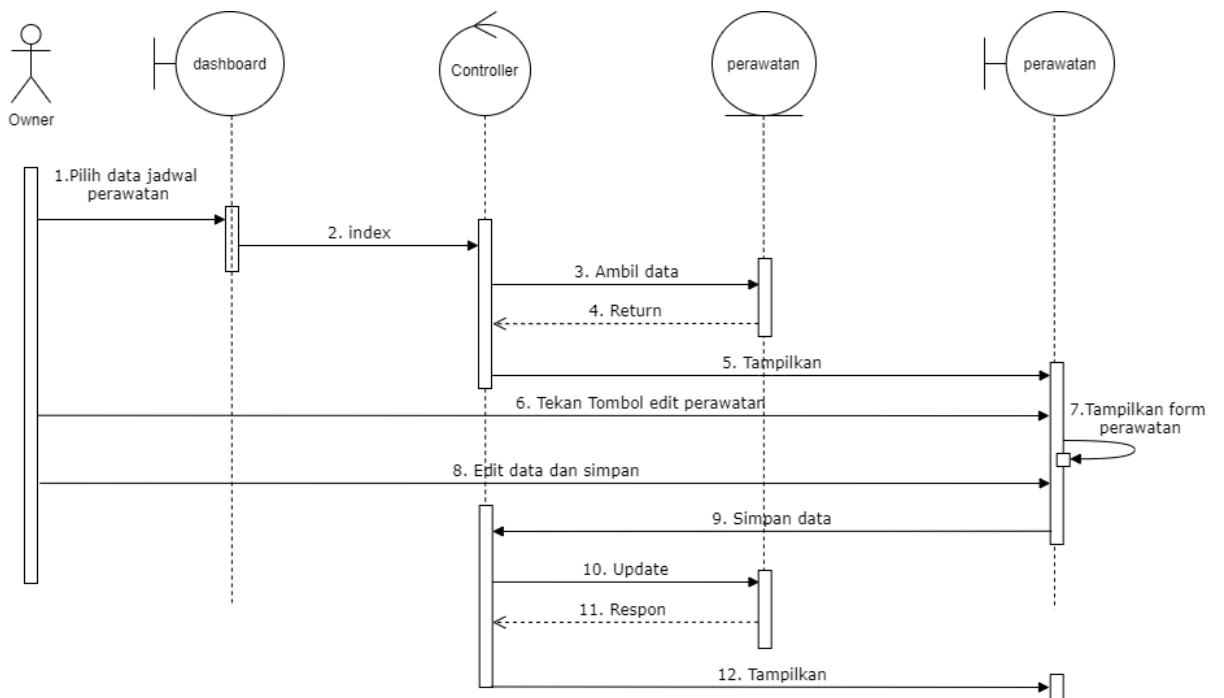
## 8. Insert perawatan



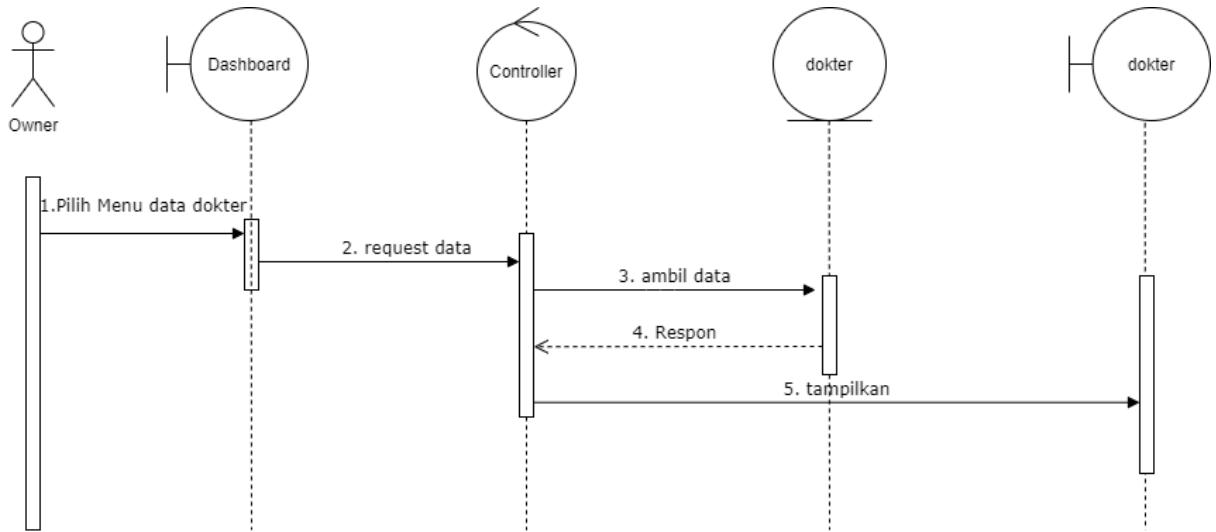
## 9. Edit pengeluaran



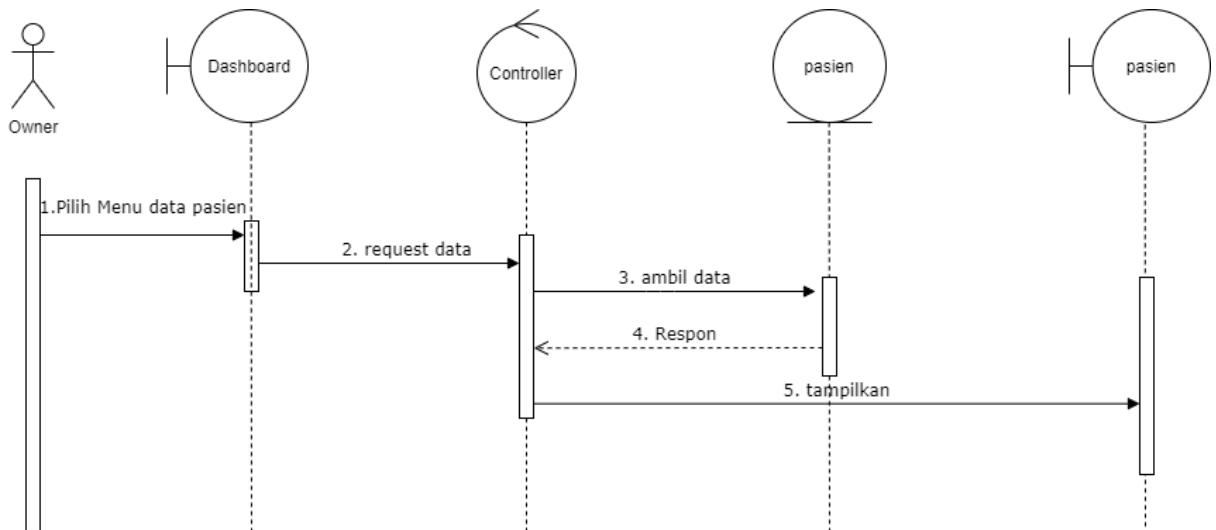
## 10. Edit Perawatan



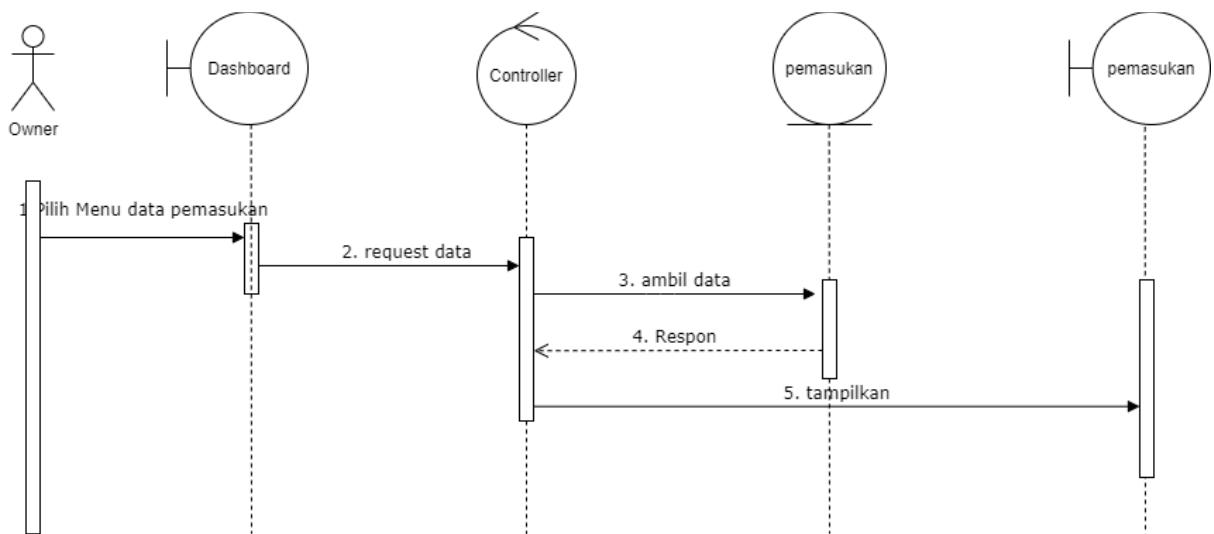
### 11. Menampilkan data dokter



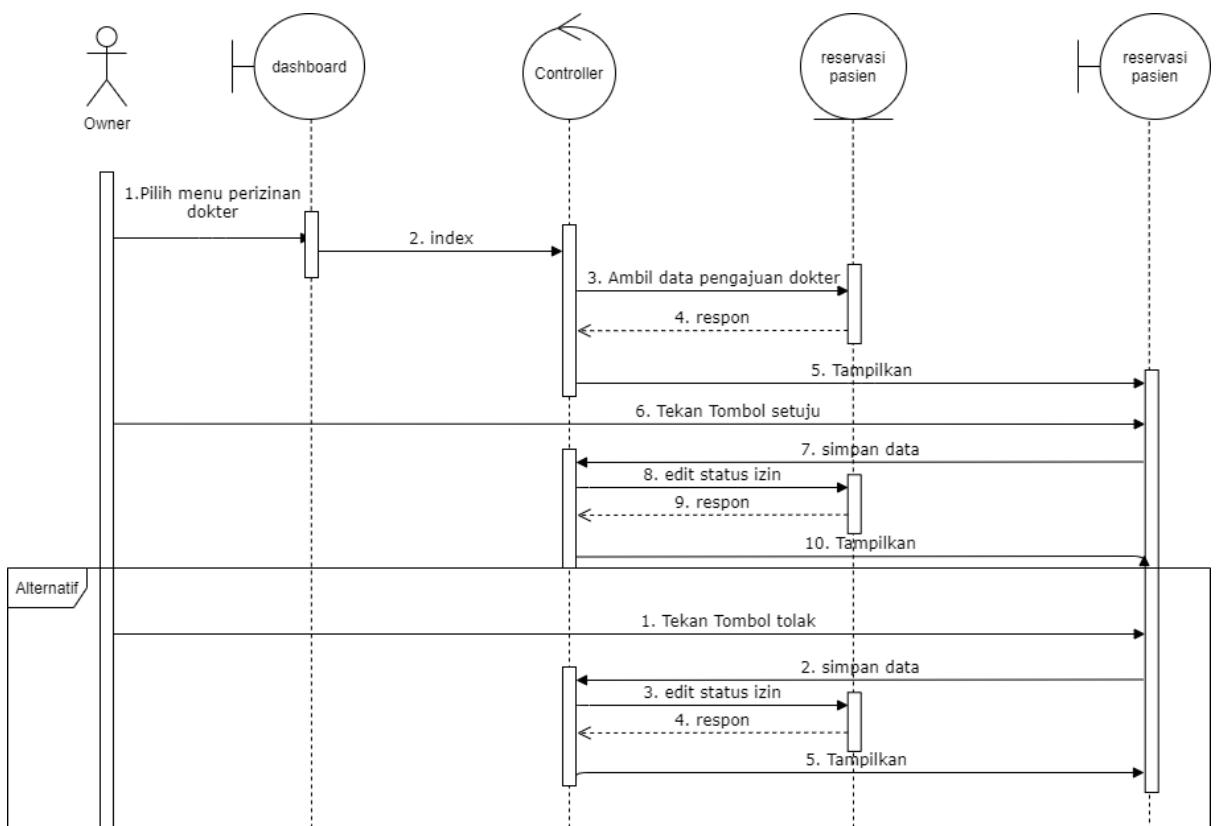
### 12. Menampilkan data pasien



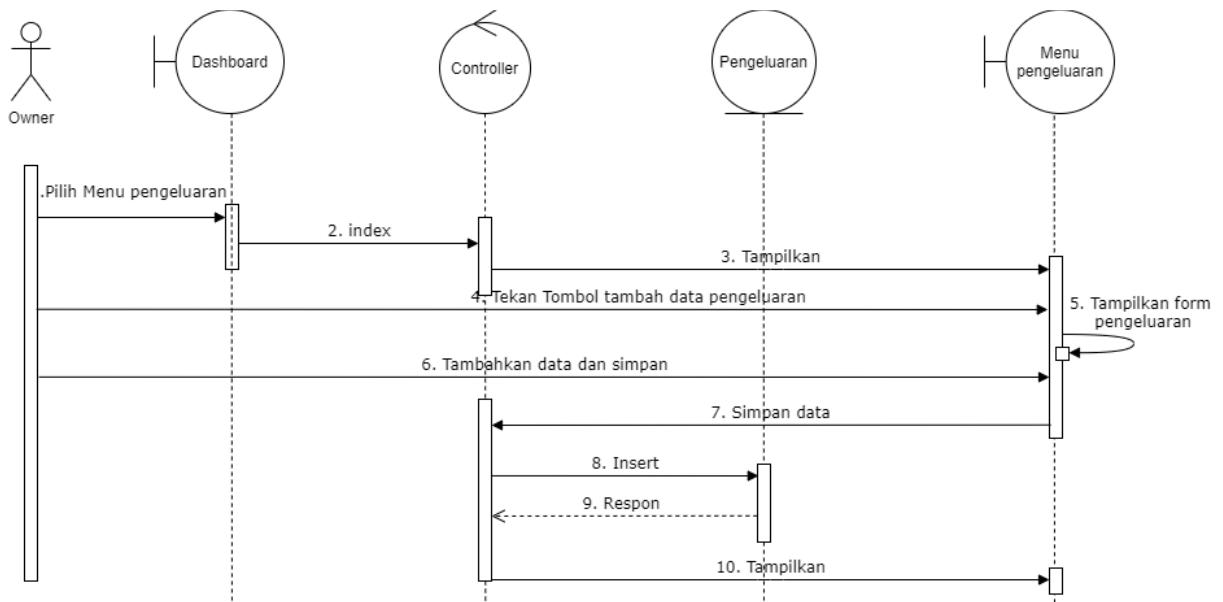
### 13. Menampilkan data pemasukan



### 14. Konfirmasi perizinan oleh owner dari dokter

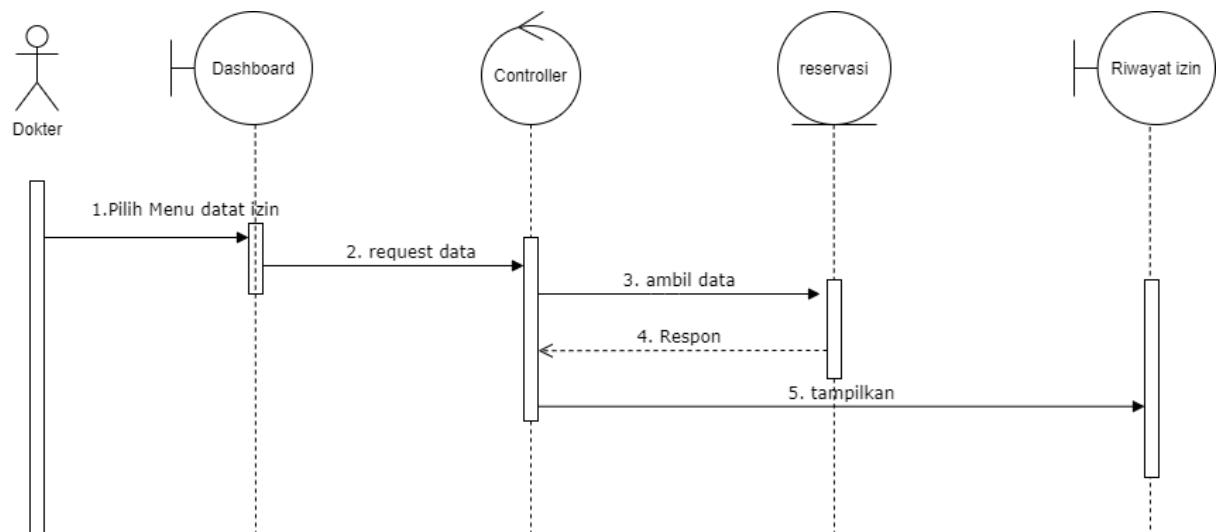


### 15. Menambahkan data pengeluaran

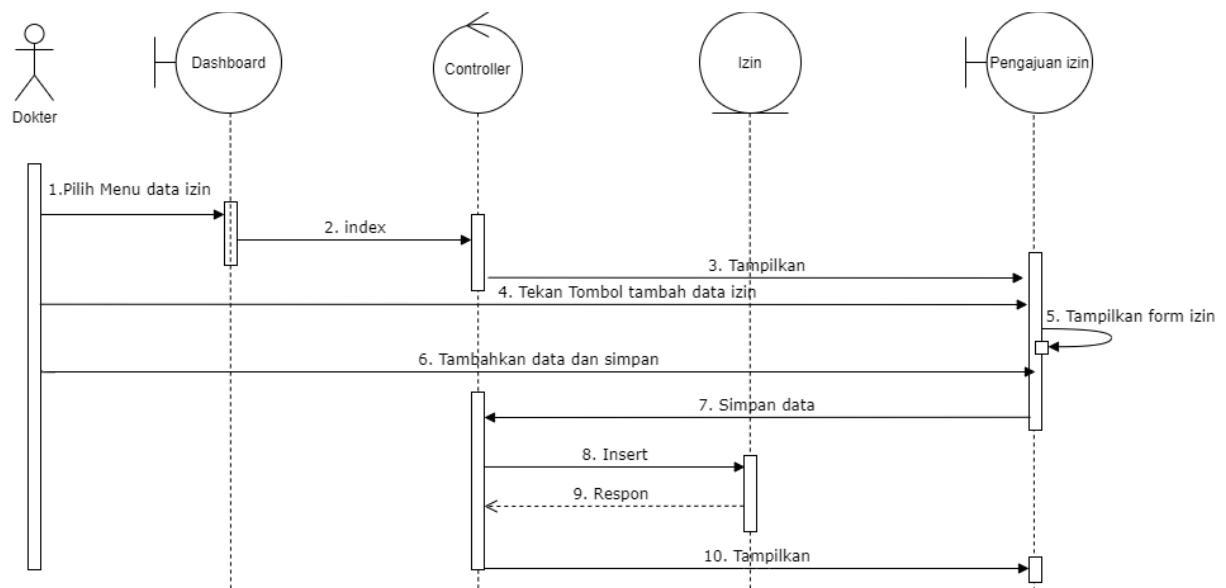


dokter

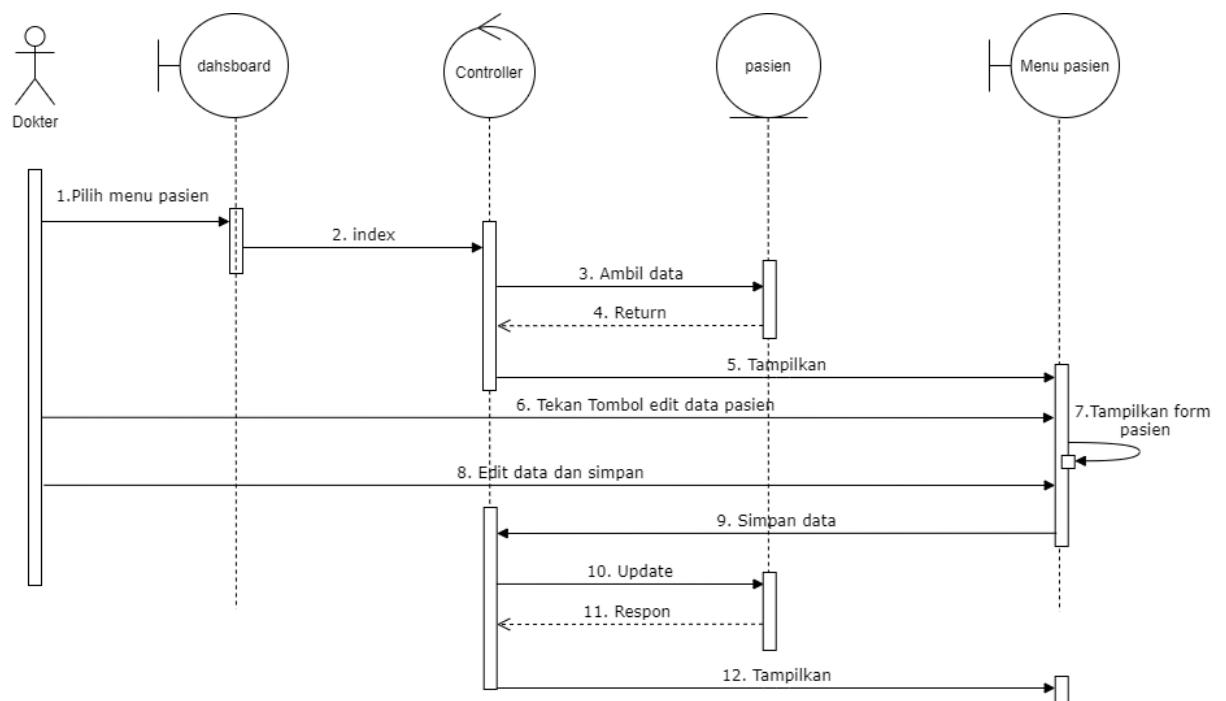
### 16. Menampilkan Riwayat izin dokter



### 17. Pengajuan izin kepada owner

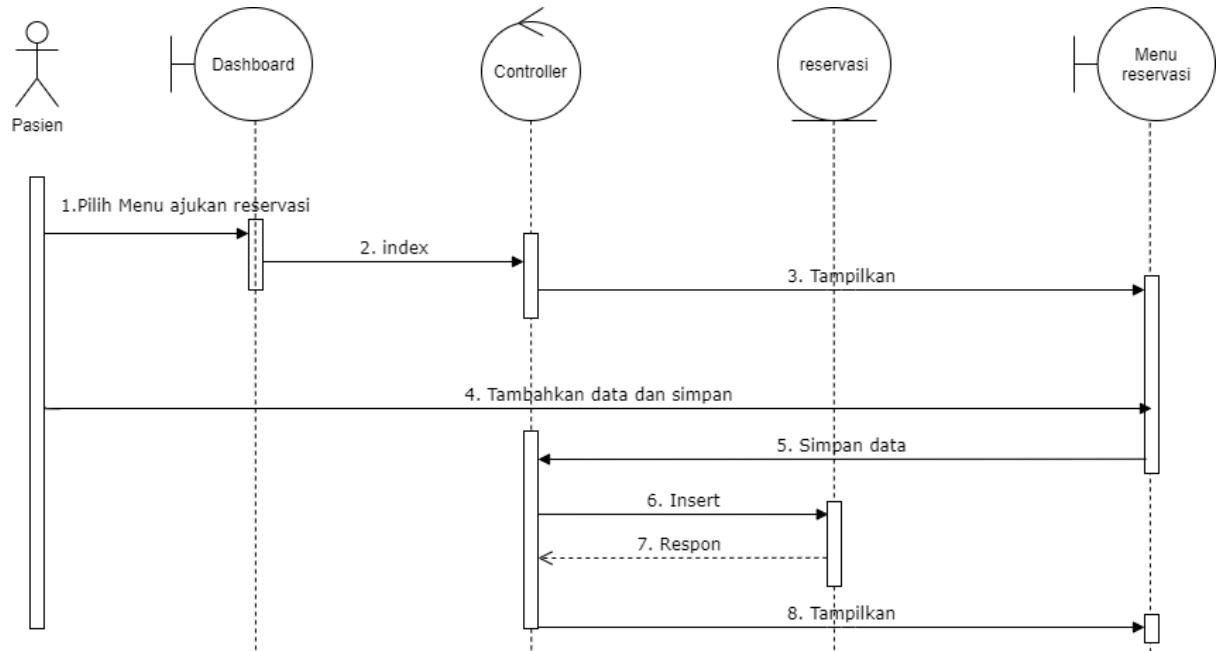


### 18. Edit data pasien

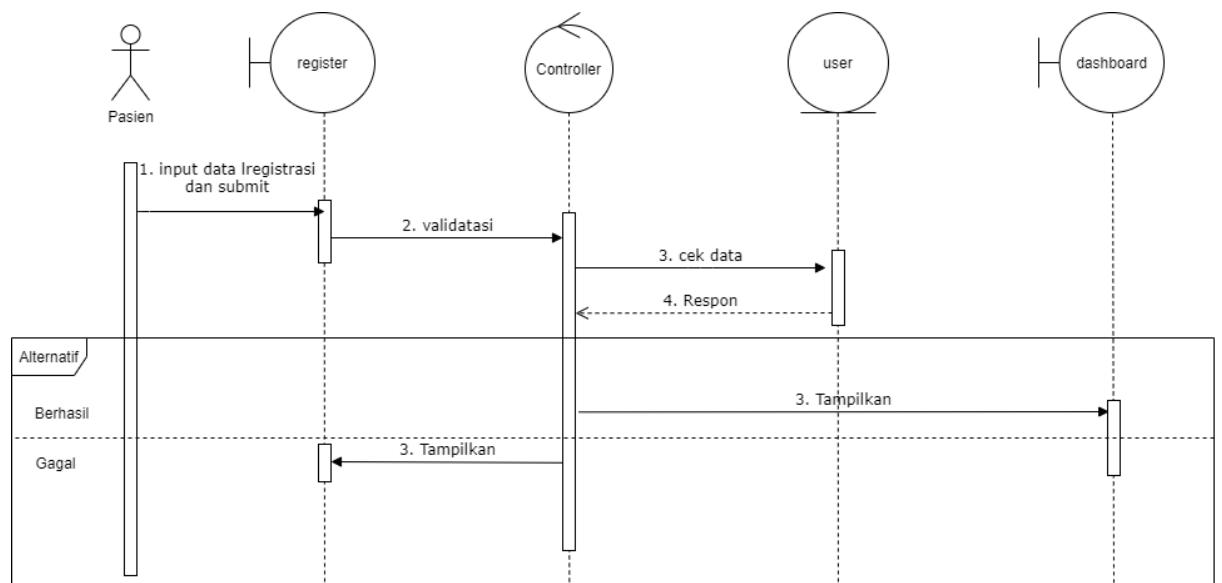


## Pasien

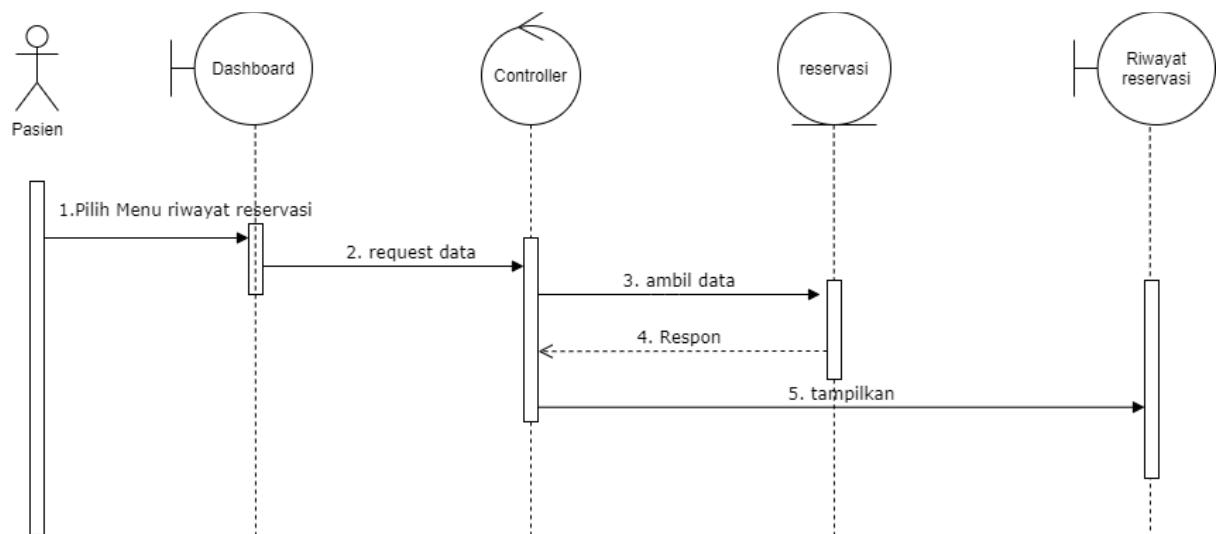
### 19. Menambahkan data reservasi



### 20. Registrasi akun pasien



## 21. Menampilkan Riwayat Reservasi

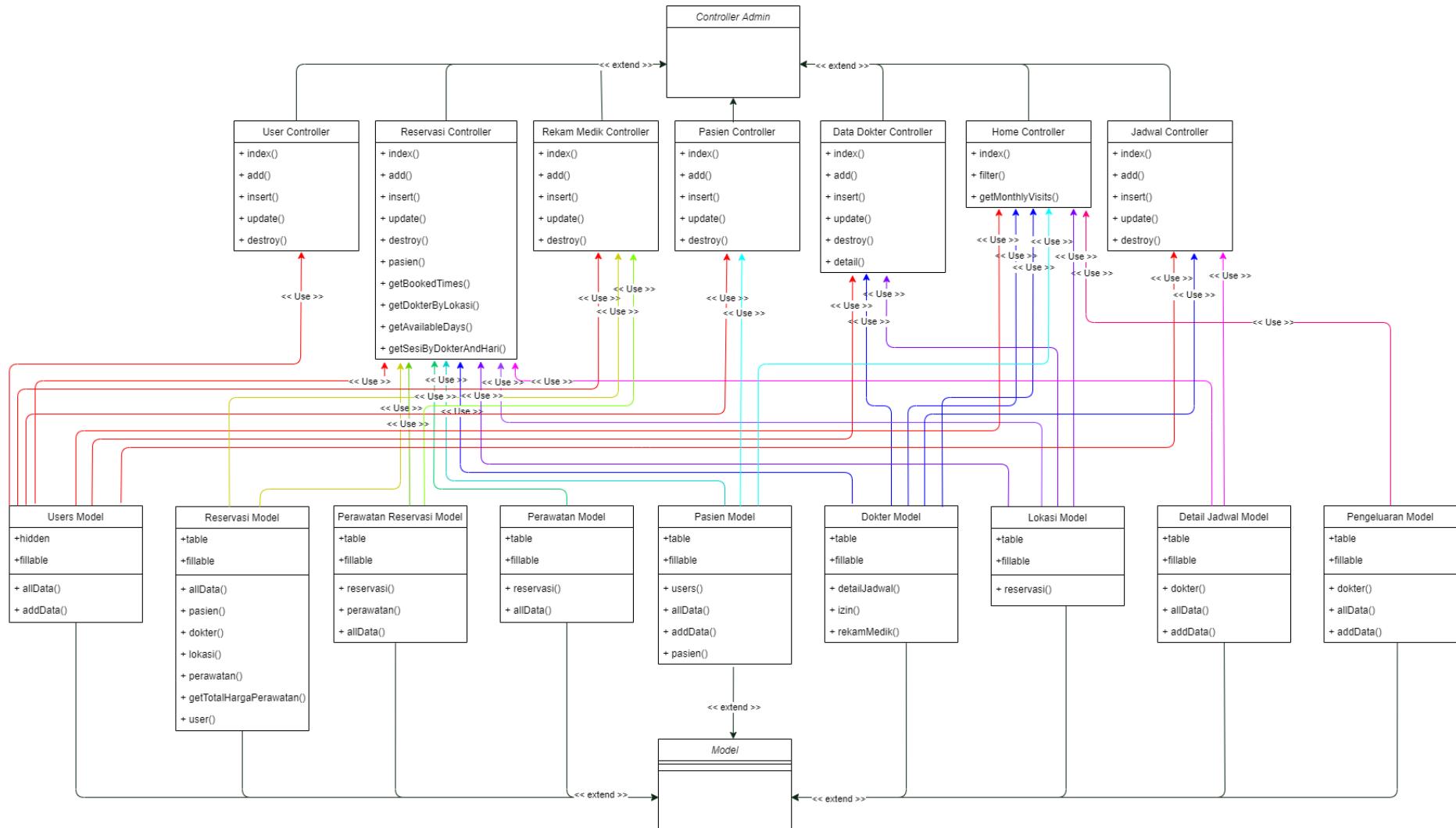


## **LAMPIRAN B**

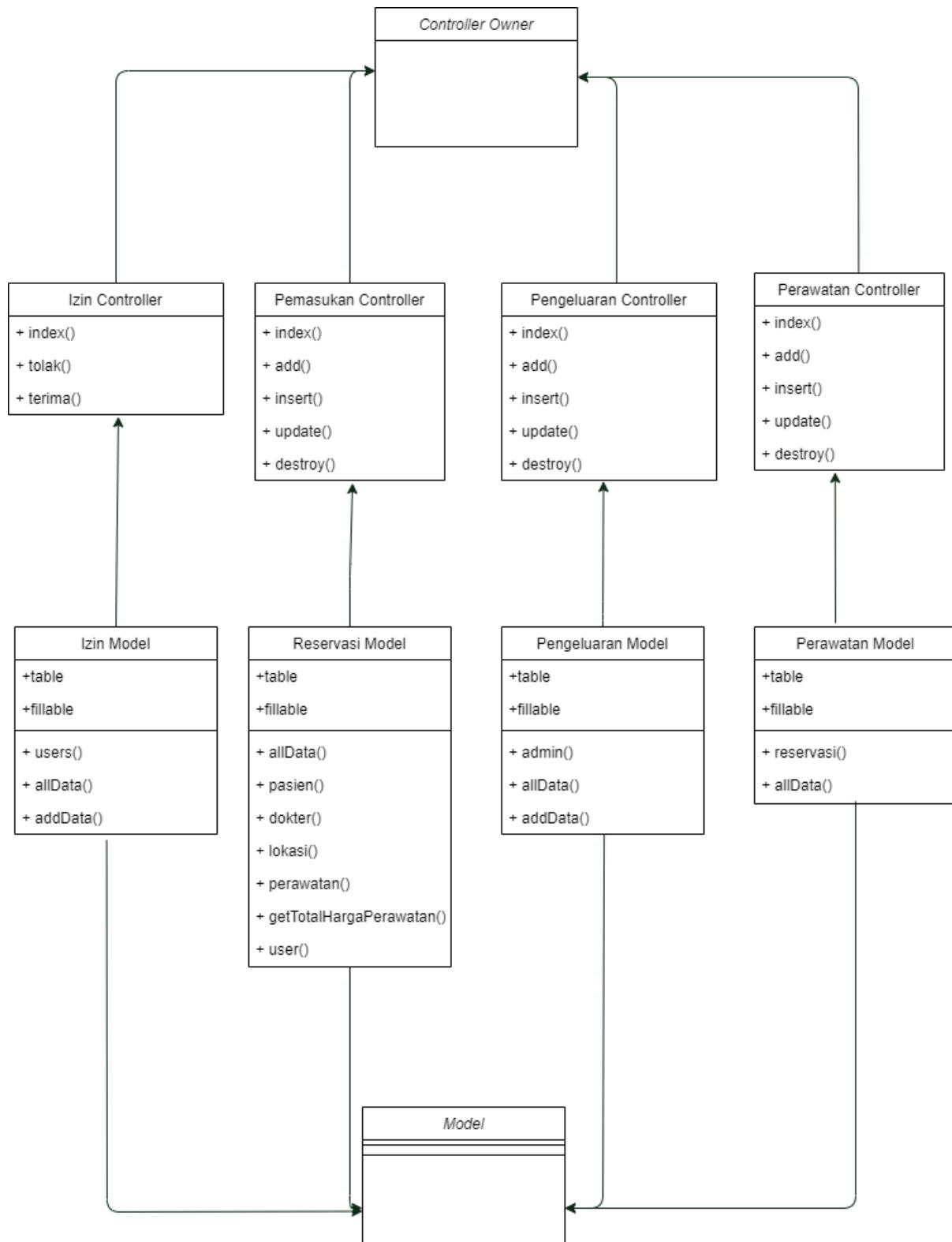
***CLASS DIAGRAM***

**(LANJUTAN)**

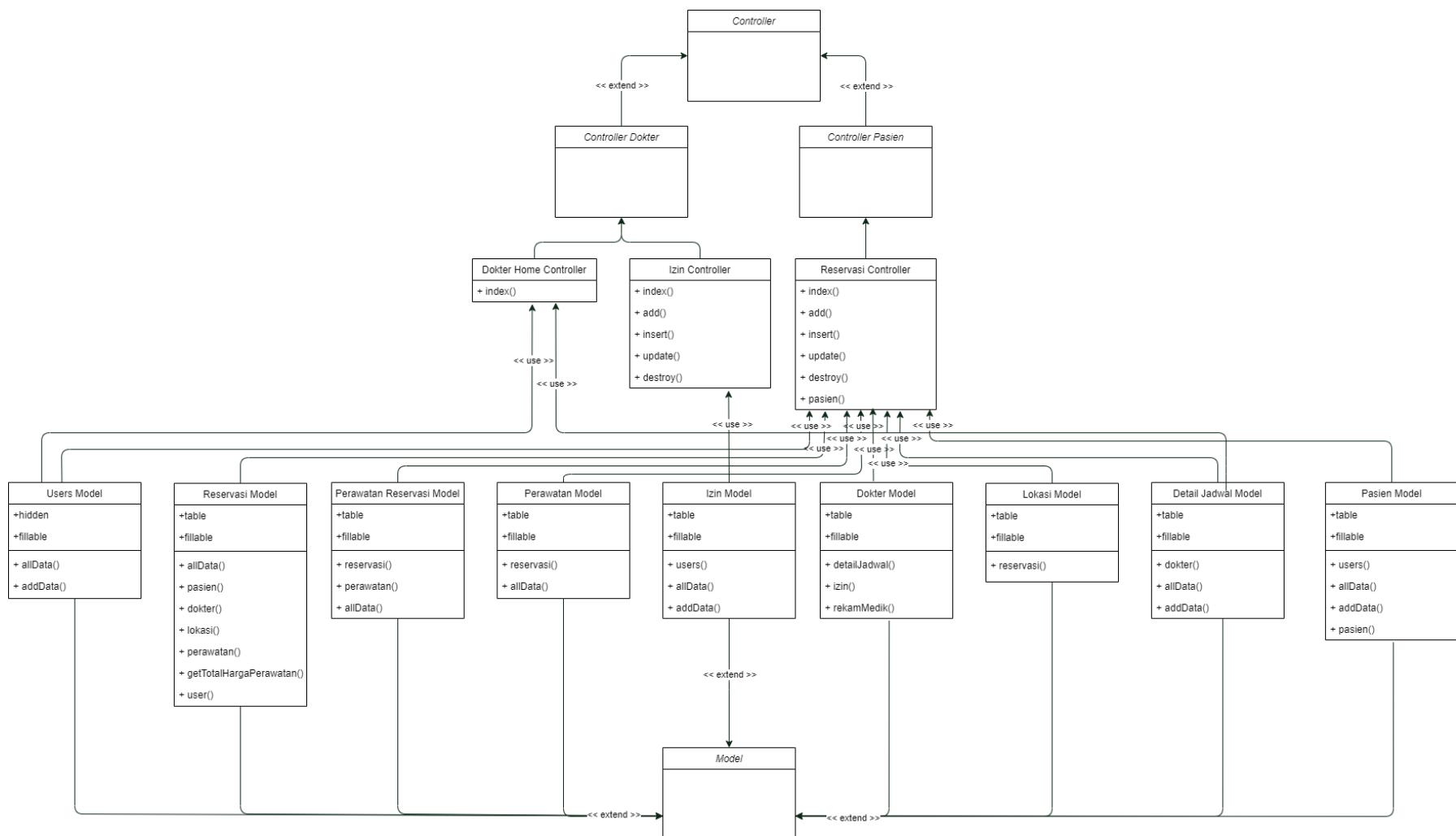
## 1. Class Diagram Admin



## 2. Class Diagram Owner



### 3. Class Diagram Pasien dan Dokter



**LAMPIRAN C**

***MOCKUP ANTARMUKA***

**(LANJUTAN)**

## 1. Mockup Antarmuka Halaman Landing Page Setelah Login

**XENON DENTAL HOUSE**

Telah lebih melayani 1000 pasien dengan memberikan berbagai keunggulan pada layanannya

keunggulan1 keunggulan2 keunggulan3 keunggulan4

**CAPTION**

**Placeholder Text:**

**LAYANAN PERAWATAN**

**DOKTER**

Column One Column Two Column Three Column Four

Twenty One  
Thirty Two  
Forty Three  
Fifty Four

Sixty Five  
Seventy Six  
Eighty Seven  
Ninety Eight

One Two  
Three Four  
Five Six  
Seven Eight

App Store Google Play

Join Us

Facebook Twitter Instagram LinkedIn

Companyname © 2023. All rights reserved.

Eleven Twelve Thirteen

## 2. Mockup Antarmuka Halaman registrasi

The mockup shows a 'Sign Up' form with the following fields:

- First Name: Placeholder
- Last Name: Placeholder
- Email: Placeholder
- Password: Placeholder

Below the fields is a blue button labeled "Button Text".

Text below the button: "Or sign up with:"

Buttons for social logins: Google, Apple, Twitter.

Text at the bottom: "Sudah punya akun? login"

## 3. Mockup Antarmuka Halaman Login

The mockup shows a 'Welcome Back' login form with the following fields:

Text above the form: "Silahkan Log In"

Email Address: Placeholder

Password: Placeholder

Text below the password field: "It must be a combination of minimum 8 letters, numbers, and symbols."

Blue button labeled "Log In".

Text below the button: "Or log in with:"

Buttons for social logins: Google, Apple, Twitter.

Text at the bottom: "No account yet? Sign Up"

#### 4. Mockup Antarmuka Halaman Profile

The mockup shows a sidebar on the left with the 'XENON DENTAL HOUSE' logo at the top. Below it is a navigation menu with the following items:

- Dashboard** (selected)
- Reservasi
- Data Pasien
- Data Dokter
- User
- Profile**

The main content area is titled 'My Account'. It contains several input fields for account information:

Name	Alif
Last Name	Abdul
Email Address	(empty)
Current Password	(empty)
New Password	(empty)
Confirm Password	(empty)

A 'Save Changes' button is located at the bottom of the form.

#### 5. Mockup Antarmuka Halaman Jadwal Dokter

The mockup shows a sidebar on the left with the 'XENON DENTAL HOUSE' logo at the top. Below it is a navigation menu with the following items:

- Jadwal** (selected)
- Data Pasien
- Data Absen

The main content area is titled 'Jadwal Saya' (My Schedule). It displays a table showing the doctor's schedule by day:

Hari	Sesi
Senin	1
Selasa	2
Rabu	1
Rabu	2
Kamis	1

Below the schedule table, there are two time blocks:

- Shift 1: 10:00 – 15:00 WIB**
- Shift 2: 15:00 – 21:00 WIB**

## 6. Mockup Antarmuka Halaman Riwayat Perawatan

The mockup shows a list of treatment history entries:

- 27/04/2024 Tambal gigi : RP. 200.000
- 20/04/2024 Tambal gigi : RP. 200.000
- 19/04/2024 Tambal gigi : RP. 200.000
- 19/04/2024 Scaling : RP. 200.000
- 01/04/2024 Scaling : RP. 200.000

## 7. Mockup Antarmuka Halaman Reservasi Pasien

The mockup shows a list of patient reservations:

No	Nama	Tanggal	Aksi
1	Alif	27/04/2024	<a href="#">Detail</a> <a href="#">Terima</a> <a href="#">Tolak</a>
2	Abdul	27/04/2024	<a href="#">Detail</a> <a href="#">Terima</a> <a href="#">Tolak</a>
3	Rauf	27/04/2024	<a href="#">Detail</a> <a href="#">Terima</a> <a href="#">Tolak</a>
4	Fajri	27/04/2024	<a href="#">Detail</a> <a href="#">Terima</a> <a href="#">Tolak</a>
5	Arif	27/04/2024	<a href="#">Detail</a> <a href="#">Terima</a> <a href="#">Tolak</a>

## 8. Mockup Antarmuka Halaman Tambah Reservasi

Pengisian Reservasi

Tempat

Padang      Bukittinggi

Nama Dokter

Jadwal

Jenis Perawatan

Nomor HP

Email

Submit

## 9. Mockup Antarmuka Halaman Data Pasien User Admin

No	Nama	No HP	Tanggal perawatan	Rekam Medik	Aksi
1	Alif	082234556781	27/04/2024	Q	Edit Hapus
2	Abdul	082234556781	27/04/2024	Q	Edit Hapus
3	Rauf	082234556781	27/04/2024	Q	Edit Hapus
4	Fajri	082234556781	27/04/2024	Q	Edit Hapus
5	Arif	082234556781	27/04/2024	Q	Edit Hapus

## 10. Mockup Antarmuka Halaman Data Pasien User Owner

The screenshot shows a mobile application interface for 'XENON DENTAL HOUSE'. At the top, there's a navigation bar with icons for notifications, messages, and a user profile labeled 'Admin A'. Below the header, the main content area is titled 'Data Pasien'. On the left, a sidebar lists several menu items: 'Dashboard', 'Data Pasien' (selected), 'Data Dokter', 'Data Pelayanan', 'Data Pengeluaran', and 'Data Pemasukan'. The main content area displays a table with five rows of patient data:

No	Nama	No HP	Tanggal perawatan
1	Alif	082234556781	27/04/2024
2	Abdul	082234556781	27/04/2024
3	Rauf	082234556781	27/04/2024
4	Fajri	082234556781	27/04/2024
5	Arif	082234556781	27/04/2024

## 11. Mockup Antarmuka Halaman

The screenshot shows a mobile application interface for 'XENON DENTAL HOUSE'. At the top, there's a navigation bar with icons for notifications, messages, and a user profile labeled 'Doktor D'. Below the header, the main content area is titled 'Rekam Medik'. On the left, a sidebar lists 'Jadwal' and 'Rekam Medik' (selected). The main content area displays patient information for 'Alif':

**Nama : Alif**  
**Tanggal lahir & Umur : 20**  
**Perawatan**  
**Riwayat**

**Kondisi Pasien**

**Lembar Tindakan :**  
**S : Gigi Berlubang**  
**O:**  
**A**  
**P**

## 12. Mockup Antarmuka Halaman edit Rekam Medik

XENON DENTAL HOUSE

Jadwal

Rekam Medik

Data Absen

### Edit Rekam Medik

Kembali

Nama

Tanggal lahir / Umur

Riwayat Penyakit / alergi

Dokter

Jenis Perawatan

Tindakan

Biaya Perawatan

Submit

## 13. Mockup Antarmuka Halaman Izin Dokter

XENON DENTAL HOUSE

Jadwal

Data Pasien

Data Absen

### Data Absen

Kembali

No	Tanggal	Alasan	Aksi
1	27/04/2024	Sakit	<button>Edit</button> <button>Hapus</button>
2	27/04/2024	Kemalangan	<button>Edit</button> <button>Hapus</button>
3	27/04/2024	Sakit	<button>Edit</button> <button>Hapus</button>
4	27/04/2024	Sakit	<button>Edit</button> <button>Hapus</button>
5	27/04/2024	Cuti	<button>Edit</button> <button>Hapus</button>

Tambahkan Data Absen

#### 14. Mockup Antarmuka Halaman Data Pasien User Dokter

No	Nama	No HP	Tanggal perawatan	Rekam Medik
1	Alif	082234556781	27/04/2024	<input type="button" value="Q"/>
2	Abdul	082234556781	27/04/2024	<input type="button" value="Q"/>
3	Rauf	082234556781	27/04/2024	<input type="button" value="Q"/>
4	Fajri	082234556781	27/04/2024	<input type="button" value="Q"/>
5	Arif	082234556781	27/04/2024	<input type="button" value="Q"/>

#### 15. Mockup Antarmuka Halaman Data Dokter User Admin

No	Nama	No HP	Jadwal	Aksi
1	Andi	082234556781	Detail jadwal	<input type="button" value="Edit"/> <input type="button" value="Hapus"/>
2	Budi	082234556781	Detail jadwal	<input type="button" value="Edit"/> <input type="button" value="Hapus"/>
3	Rauf	082234556781	Detail jadwal	<input type="button" value="Edit"/> <input type="button" value="Hapus"/>
4	Fajri	082234556781	Detail jadwal	<input type="button" value="Edit"/> <input type="button" value="Hapus"/>
5	Arif	082234556781	Detail jadwal	<input type="button" value="Edit"/> <input type="button" value="Hapus"/>

[Tambahkan Data Dokter](#)

## 16. Mockup Antarmuka Halaman Data Dokter User Owner

**XENON DENTAL HOUSE**

**Data Dokter**

No	Nama	No HP	Jadwal
1	Andi	082234556781	<a href="#">Detail jadwal</a>
2	Budi	082234556781	<a href="#">Detail jadwal</a>
3	Rauf	082234556781	<a href="#">Detail jadwal</a>
4	Fajri	082234556781	<a href="#">Detail jadwal</a>
5	Arif	082234556781	<a href="#">Detail jadwal</a>

## 17. Mockup Antarmuka Halaman Dahsboard Dokter

**XENON DENTAL HOUSE**

**Jadwal Dokter Nama**

Hari	Sesi	Aksi
Senin	1	<a href="#">Edit</a> <a href="#">Hapus</a>
Selasa	2	<a href="#">Edit</a> <a href="#">Hapus</a>
Rabu	1	<a href="#">Edit</a> <a href="#">Hapus</a>
Rabu	2	<a href="#">Edit</a> <a href="#">Hapus</a>
Kamis	1	<a href="#">Edit</a> <a href="#">Hapus</a>

[Tambahkan jadwal dokter](#)

**Shift 1: 10:00 – 15:00 WIB**

**Shift 2: 15:00 – 21:00 WIB**

## 18. Mockup Antarmuka Halaman Rekam Medik User Dokter

The mockup shows a mobile application interface for a dental clinic. The top navigation bar includes a bell icon, an envelope icon, and a 'Doktor' profile section with a letter 'D'. The left sidebar, titled 'XENON DENTAL HOUSE', lists menu items: Dashboard, Reservasi, Data Pasien (selected), Data Dokter, User, and Profile. The main content area is titled 'Rekam Medik' and displays patient information: Nama : Alif, Tanggal lahir & Umur : 20, Perawatan, Riwayat. Below this is a section titled 'Kondisi Pasien' with the text: Lembar Tindakan : S : Gigi Berlubang, O:, A, P. At the bottom of the content area is an 'Edit' button.

## 19. Mockup Antarmuka Halaman Edit data pasien User Dokter

The mockup shows a mobile application interface for a dental clinic. The top navigation bar includes a bell icon, an envelope icon, and a 'Doktor' profile section with a letter 'D'. The left sidebar, titled 'XENON DENTAL HOUSE', lists menu items: Dashboard, Reservasi, Data Pasien (selected), Data Dokter, User, and Profile. The main content area is titled 'Edit Data Pasien' and contains form fields for editing patient data: Nama, Nomor HP, Alamat, Email, and Tanggal Perawatan. At the bottom of the content area is a 'Submit' button.

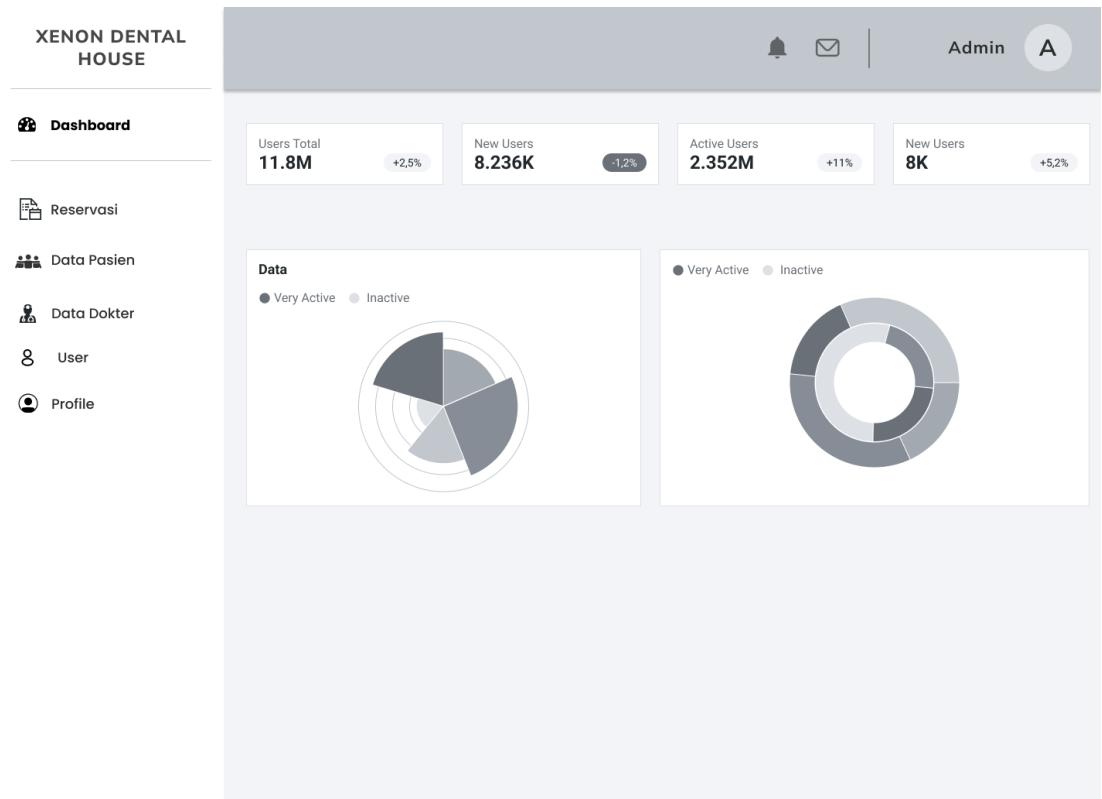
## 20. Mockup Antarmuka Halaman Tambah jadwal dokter

The screenshot shows a mobile application interface for 'XENON DENTAL HOUSE'. The top navigation bar includes icons for notifications, messages, and a user profile labeled 'Doktor D'. The left sidebar menu lists several options: 'Dashboard', 'Reservasi', 'Data Pasien', 'Data Dokter', 'User', and 'Profile'. The main content area is titled 'Tambahkan Jadwal Dokter' (Add Doctor Schedule). It contains two input fields: 'Hari' (Day) and 'Sesi' (Session), both with dropdown menus. A 'Submit' button is located below these fields. In the bottom right corner of the main content area, there are two time ranges: 'Shift 1: 10:00 – 15:00 WIB' and 'Shift 2: 15:00 – 21:00 WIB'. A 'Kembali' (Back) button is positioned in the top right corner of the main content area.

## 21. Mockup Antarmuka Halaman Edit Data dokter

The screenshot shows a mobile application interface for 'XENON DENTAL HOUSE'. The top navigation bar includes icons for notifications, messages, and a user profile labeled 'Doktor D'. The left sidebar menu lists several options: 'Jadwal', 'Rekam Medik', and 'Data Absen'. The main content area is titled 'Edit Data Dokter' (Edit Doctor Data). It contains four input fields: 'Nama' (Name), 'Nomor Handphone' (Phone Number), and 'Email', each with a dropdown menu, followed by a 'Submit' button. A 'Kembali' (Back) button is positioned in the top right corner of the main content area.

## 22. Mockup Antarmuka Halaman Dashboard Admin



## 23. Mockup Antarmuka Halaman Tambah data pelayanan

The form has a header with the logo 'XENON DENTAL HOUSE' and user status indicators for 'Doktor' and 'D'. It includes a back button 'Kembali'. The main section is titled 'Tambahkan Data Pelayanan' and contains fields for 'Nama Pelayanan' and 'Harga', with a 'Submit' button at the bottom.

## 24. Mockup Antarmuka Halaman Data Pelayanan

**XENON DENTAL HOUSE**

**Harga Pelayanan**

No	Pelayanan	Harga	Aksi
1	Cabut Gigi	200.000	Edit Hapus
2	Scaling	200.000	Edit Hapus
3	Bleaching	100.000	Edit Hapus
4	Gigi Tiruan	200.000	Edit Hapus
5	Tambal Gigi	1000.000	Edit Hapus

**Tambahkan Data Pelayanan**

## 25. Mockup Antarmuka Halaman Pemasukan

**XENON DENTAL HOUSE**

**Pemasukan**

Pemasukan Bulan Ini <b>RP 100.000</b>	Pengeluaran Bulan Ini <b>RP 100.000</b>	Profit Bulan Ini <b>RP 100.000</b>
--	--	---------------------------------------

No	Pemasukan	Tanggal	Profit
1	Cabut Gigi	200.000	Edit Hapus
2	Scaling	200.000	Edit Hapus
3	Bleaching	100.000	Edit Hapus
4	Gigi Tiruan	200.000	Edit Hapus
5	Tambal Gigi	1000.000	Edit Hapus

**Tambahkan Data Pemasukan**

## 26. Mockup Antarmuka Halaman Edit Pemasukan

XENON DENTAL HOUSE

Dashboard

Data Pasien

Data Dokter

Data Pelayanan

Data Pengeluaran

Data Pemasukan

Edit Data Pemasukan

Nama Pelayanan

Harga

Submit

Kembali

## 27. Mockup Antarmuka Halaman Data Pengeluaran

XENON DENTAL HOUSE

Dashboard

Data Pasien

Data Dokter

Data Pelayanan

Data Pengeluaran

Data Pemasukan

Pengeluaran

Pemasukan Bulan Ini  
RP 100.000

Pengeluaran Bulan Ini  
RP 100.000

Profit Bulan Ini  
RP 100.000

No	Pengeluaran	Tanggal	Action
1	Cabut Gigi	200.000	Edit Hapus
2	Scaling	200.000	Edit Hapus
3	Bleaching	100.000	Edit Hapus
4	Gigi Tiruan	200.000	Edit Hapus
5	Tambal Gigi	1000.000	Edit Hapus

Tambahkan Data Pelayanan

## 28. Mockup Antarmuka Halaman Tambah Pengeluaran

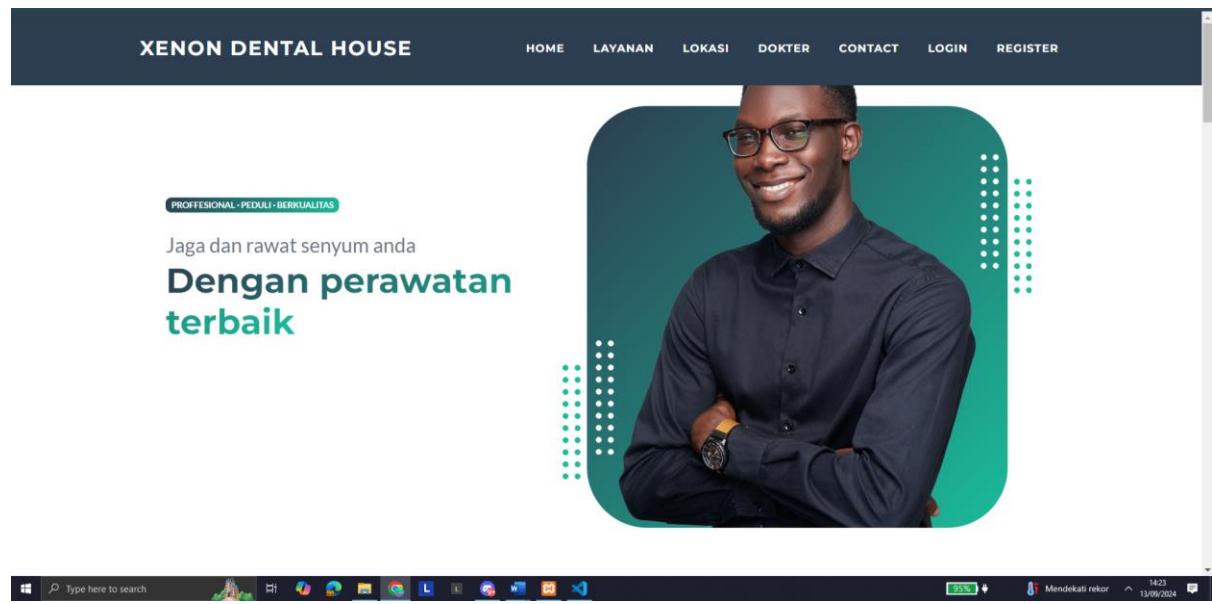
The mockup shows a user interface for adding expense data. At the top, there is a header bar with the logo 'XENON DENTAL HOUSE' and navigation icons for Dashboard, Data Pasien, Data Dokter, Data Pelayanan, Data Pengeluaran, and Data Pemasukan. On the right side of the header are icons for notifications, messages, and a user profile labeled 'Doktor D'. The main content area has a title 'Tambah data Pengeluaran' and two input fields: 'Nama Pelayanan' and 'Jumlah Pengeluaran'. A 'Submit' button is located below the input fields. In the top right corner of the content area, there is a 'Kembali' (Back) button.

**LAMPIRAN D**

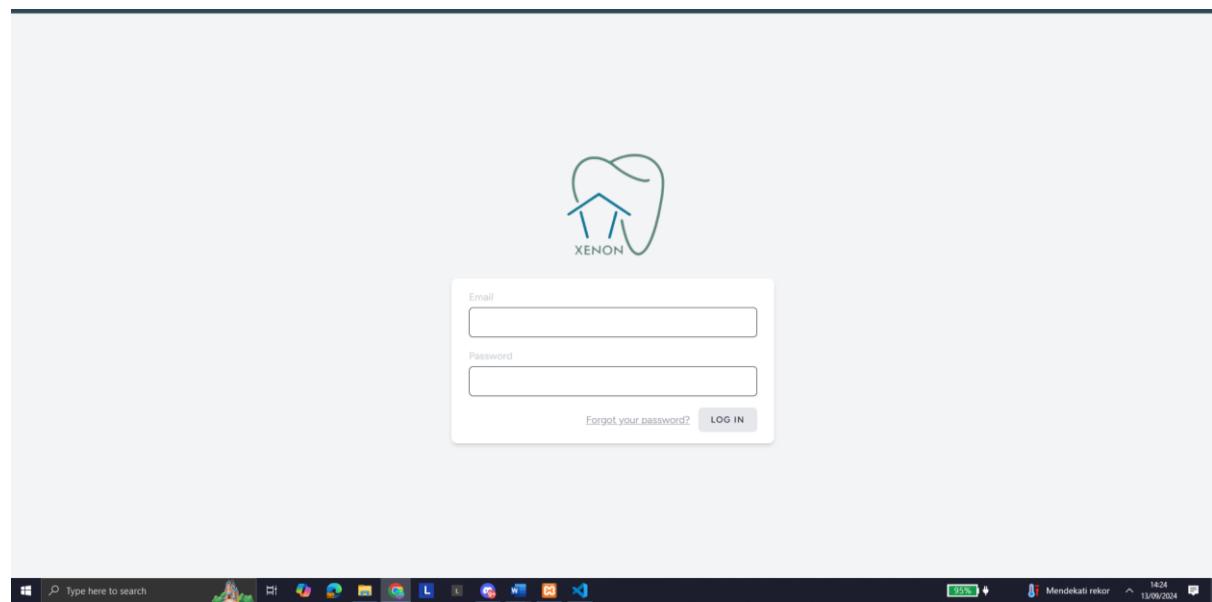
**IMPLEMENTASI ANTARMUKA**

**(LANJUTAN)**

## 1. Implementasi Antarmuka Landing Page



## 2. Implementasi Antarmuka Login User



### 3. Implementasi Antarmuka Ubah Password dan Hapus Akun User

The screenshot shows a user interface for account management. At the top, there is a section titled "Update Password" with instructions: "Ensure your account is using a long, random password to stay secure." It contains three input fields: "Current Password", "New Password", and "Confirm Password", followed by a "SAVE" button. Below this is a section titled "Delete Account" with a warning: "Once your account is deleted, all of its resources and data will be permanently deleted. Before deleting your account, please download any data or information that you wish to retain." It features a single "DELETE ACCOUNT" button.

### 4. Implementasi Antarmuka Ubah Informasi Profile

The screenshot shows a user interface for profile management. At the top, it says "Profile". Below it is a section titled "Profile Information" with the sub-instruction: "Update your account's profile information and email address." It has two input fields: "Name" (containing "rauf") and "Email" (containing "rauf@gmail.com"), followed by a "SAVE" button. Further down is another section titled "Update Password" with the same security instruction as the previous screenshot. It includes "Current Password" and "New Password" input fields.

## 5. Implementasi Antarmuka Tambah Reservasi (User Admin)

Form Tambah Reservasi

Nama:

Tempat Lahir:

Tanggal Lahir:  dd/mm/yyyy

Pekerjaan:

Alamat:

No Telepon:

Perawatan (Pilih maksimal 2):

Lokasi:  Pilih Lokasi

## 6. Implementasi Antarmuka Data Pasien (User Admin)

Data Pasien

no	Nama	Pekerjaan	Alamat	No HP	Rekam Medik	Aksi
1	Budi Santoso	PNS	Jl. Merdeka No.1, Jakarta	089617702747	Rekam Medik	<button>Edit</button> <button>Hapus</button>
2	Siti Aminah	Guru	Jl. Asia Afrika No.2, Bandung	082288051901	Rekam Medik	<button>Edit</button> <button>Hapus</button>
3	Robert	Kerja	jalan budi utomo	085348923	Rekam Medik	<button>Edit</button> <button>Hapus</button>
4	tono	mahasiswa	jalan m	082288051901	Rekam Medik	<button>Edit</button> <button>Hapus</button>
5	dimas	dokter	jalan agung	082288051901	Rekam Medik	<button>Edit</button> <button>Hapus</button>
6	alif abdul	mahasiswa	pauh	082288051901	Rekam Medik	<button>Edit</button> <button>Hapus</button>
7	dimas	mahasiswa	pauh	082288051901	Rekam Medik	<button>Edit</button> <button>Hapus</button>

Showing 1 to 7 of 7 entries

Copyright © Xenon Dental House

## 7. Implementasi Antarmuka Tambah Data Pasien (User Admin)

Tambah data Pasien

Nama Pasien  
Siti Aminah

Tempat Lahir  
Bandung

Tanggal Lahir  
1985-08-24

Pekerjaan  
Guru

Alamat  
Jl. Asia Afrika No.2, Bandung

No HP  
082288051901

**Simpan**

Copyright © Xenon Dental House

## 8. Implementasi Antarmuka Rekam Medik (User Admin)

Rekam Medik Siti Aminah	
Dokter	Alif
Golongan darah	AB
Tekanan darah	44
Penyakit Jantung	Tidak Ada
Diabetes	Tidak Ada
Hepatitis	Tidak Ada
Penyakit lainnya	Tidak Ada
Alergi Makanan	Tidak Ada
Alergi Obat	Tidak Ada
Keluhan	Tidak Ada
Perawatan	Behel
Gigi	22

**Edit Rekam Medik**

Copyright © Xenon Dental House

## 9. Implementasi Antarmuka Edit Rekam Medik (User Admin)

abdu

Golongan Darah  
AB

Tekanan Darah  
44

Penyakit Jantung  
Tidak Ada

Diabetes  
Tidak Ada

Hepatitis  
Tidak Ada

Penyakit Lainnya  
Tidak Ada

Alergi makanan  
Tidak Ada

Alergi Obat  
Tidak Ada

## 10. Implementasi Antarmuka Data Dokter (User Admin)

abdu

Search: \_\_\_\_\_

no	Nama	No HP	Jadwal	Aksi
1	Dr. Andi Wijayaa	6283456789012	Detail Jadwal	<a href="#">Edit</a> <a href="#">Hapus</a>
2	Alif	6282276635819	Detail Jadwal	<a href="#">Edit</a> <a href="#">Hapus</a>
3	Name	628235567718	Detail Jadwal	<a href="#">Edit</a> <a href="#">Hapus</a>
4	Alif R	628456672712	Detail Jadwal	<a href="#">Edit</a> <a href="#">Hapus</a>
5	Bude	62845277462	Detail Jadwal	<a href="#">Edit</a> <a href="#">Hapus</a>
6	Darib	6285456723656	Detail Jadwal	<a href="#">Edit</a> <a href="#">Hapus</a>
7	Andi	6287542664643	Detail Jadwal	<a href="#">Edit</a> <a href="#">Hapus</a>

Showing 1 to 7 of 7 entries

Previous 1 Next

Copyright © Xenon Dental House

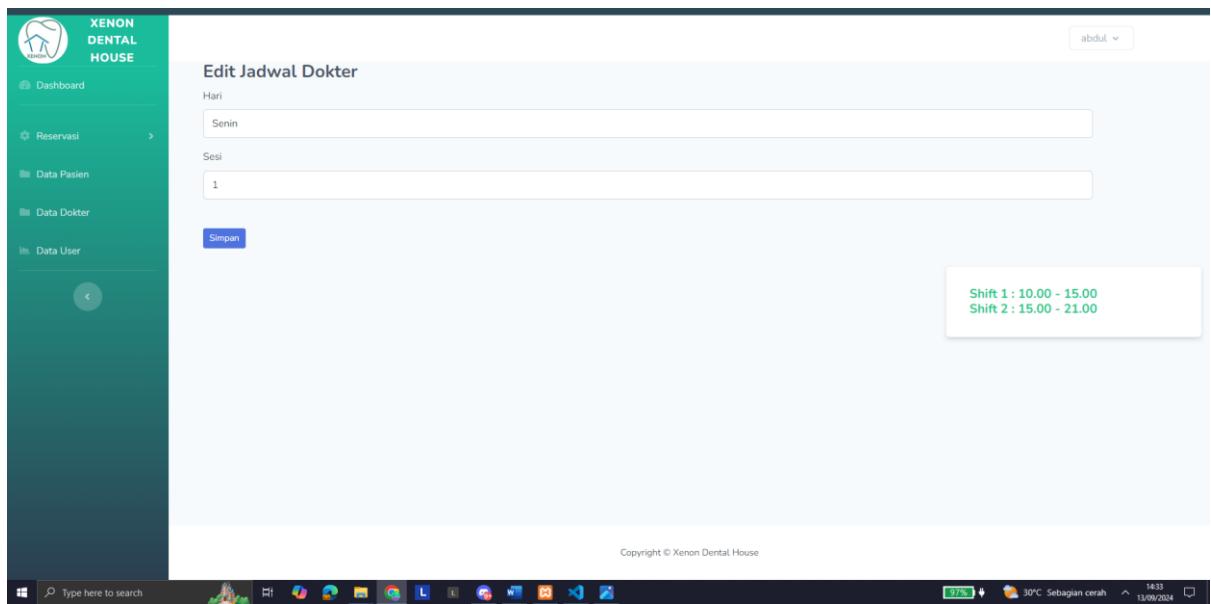
## 11. Implementasi Antarmuka Edit Data Dokter (User Admin)

The screenshot shows a web-based application interface for Xenon Dental House. On the left, there is a sidebar with a dark green gradient background containing navigation links: Dashboard, Reservasi, Data Pasien, Data Dokter, and Data User. The main content area has a white background and features a form titled "Edit data Dokter". It contains three input fields: "Nama Dokter" with the value "Dr. Andi Wijayaa", "Alamat" with the value "Jl. Kesehatan No.3, Surabaya", and "Nomor HP" with the value "083456789012". Below these fields is a blue "Simpan" button. At the top right of the main area, there is a user dropdown menu showing "abdul". At the bottom right of the main area, it says "Copyright © Xenon Dental House". The bottom of the screen shows a Windows taskbar with various icons and system status information.

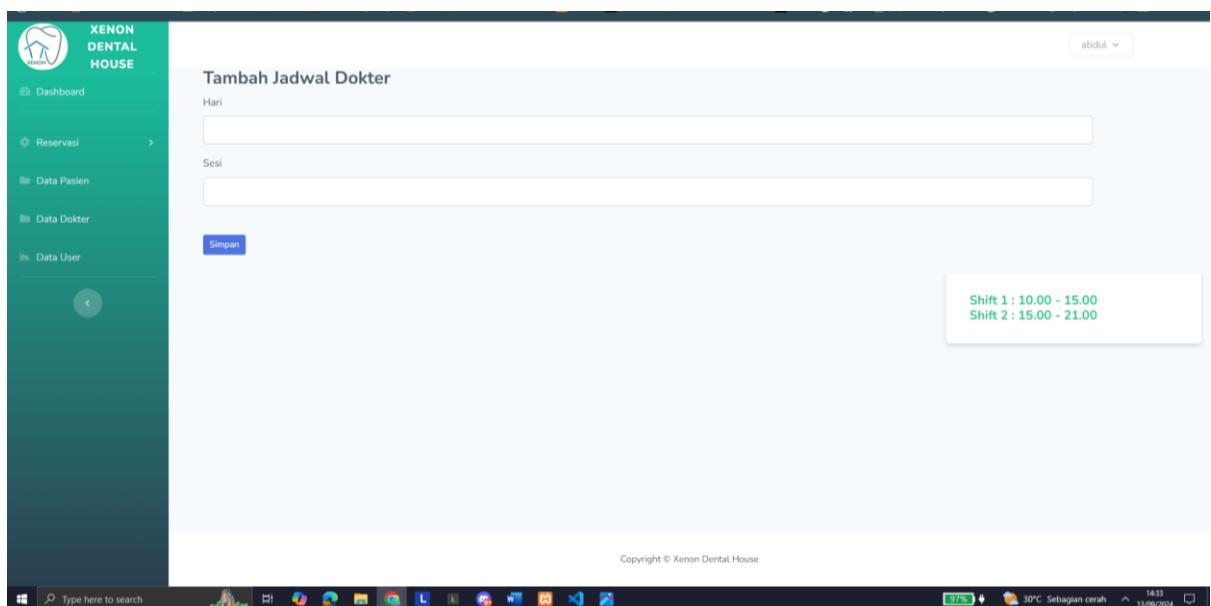
## 12. Implementasi Antarmuka Detail Jadwal Dokter (User Admin)

The screenshot shows the "Detail Jadwal Dokter" page. The sidebar on the left is identical to the previous screenshot. The main content area has a white background and features a table titled "Detail Jadwal Dokter". The table has columns: "no", "Hari", "Sesi", and "Aksi". There is one row of data: "1", "Senin", "1", and two buttons: "Edit" and "Hapus". Below the table, it says "Showing 1 to 1 of 1 entries". At the bottom left is a green "Tambahkan Data Jadwal Dokter" button. On the right, there is a message box with the text "Shift 1 : 10.00 - 15.00" and "Shift 2 : 15.00 - 21.00". At the bottom right of the main area, it says "Copyright © Xenon Dental House". The bottom of the screen shows a Windows taskbar with various icons and system status information.

### 13. Implementasi Antarmuka Edit Jadwal Dokter (User Admin)



### 14. Implementasi Antarmuka Tambah Jadwal Dokter (User Admin)



## 15. Implementasi Antarmuka Data User (User Admin)

The screenshot shows the Xenon Dental House application interface. On the left, there is a sidebar with a dark green gradient background containing navigation links: Dashboard, Reservasi, Data Pasien, Data Dokter, and Data User. The main content area has a white background and displays a table titled "Data User". The table has columns: no, Nama, Email, Role, and Aksi (Actions). The data in the table is as follows:

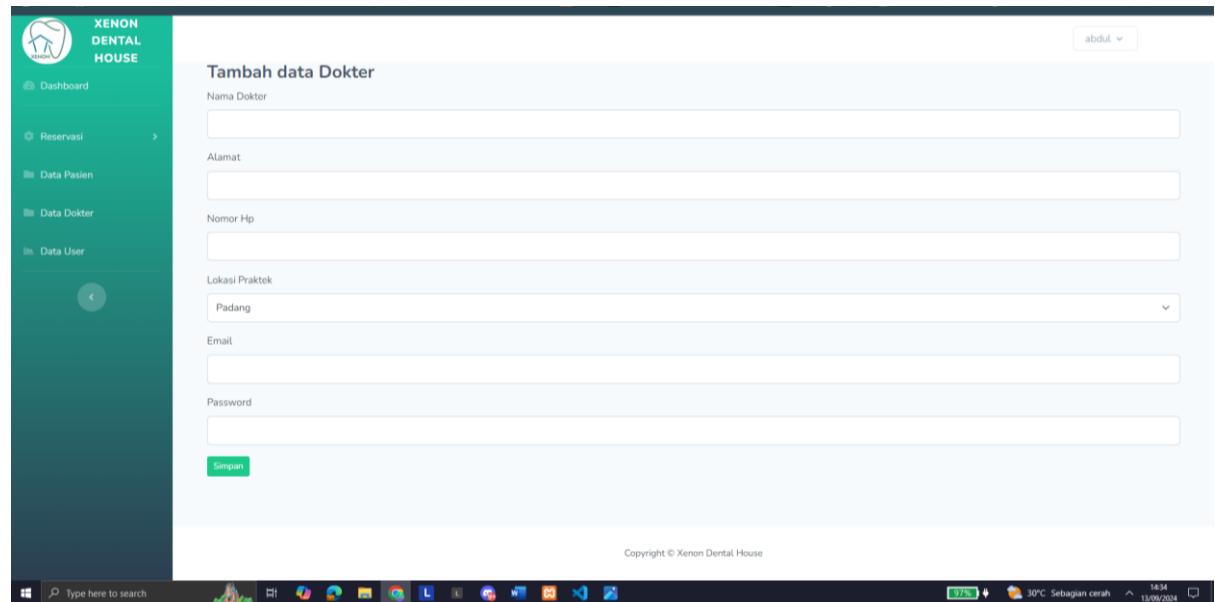
no	Nama	Email	Role	Aksi
1	alif	alifabdulrauf@gmail.com	admin	<button>Edit</button> <button>Hapus</button>
2	abdul	abdul@gmail.com	admin	<button>Edit</button> <button>Hapus</button>
3	rauf	rauf@gmail.com	owner	<button>Edit</button> <button>Hapus</button>
4	alif	alif@gmail.com	Dokter	<button>Edit</button> <button>Hapus</button>
5	Bobo	bobobobo@gmail.com	Pasien	<button>Edit</button> <button>Hapus</button>
6	Robert	robert@gmail.com	Pasien	<button>Edit</button> <button>Hapus</button>
7	Badu	badu@gmail.com	admin	<button>Edit</button> <button>Hapus</button>
8	Alif R	alifr@gmail.com	Dokter	<button>Edit</button> <button>Hapus</button>

Below the table, it says "Showing 1 to 8 of 15 entries". At the bottom, there are two green buttons: "Tambahkan Data User" and "Tambah User Dokter". The status bar at the bottom right shows "Copyright © Xenon Dental House" and the system date and time.

## 16. Implementasi Antarmuka Tambah Data User (User Admin)

The screenshot shows the Xenon Dental House application interface. On the left, there is a sidebar with a dark green gradient background containing navigation links: Dashboard, Reservasi, Data Pasien, Data Dokter, and Data User. The main content area has a white background and displays a form titled "Tambah data User". The form fields are: Name (Nama), Email, Password, and Role. Below the fields is a green "Submit" button. The status bar at the bottom right shows "Copyright © Xenon Dental House" and the system date and time.

## 17. Implementasi Antarmuka Tambah Data Dokter (User Admin)

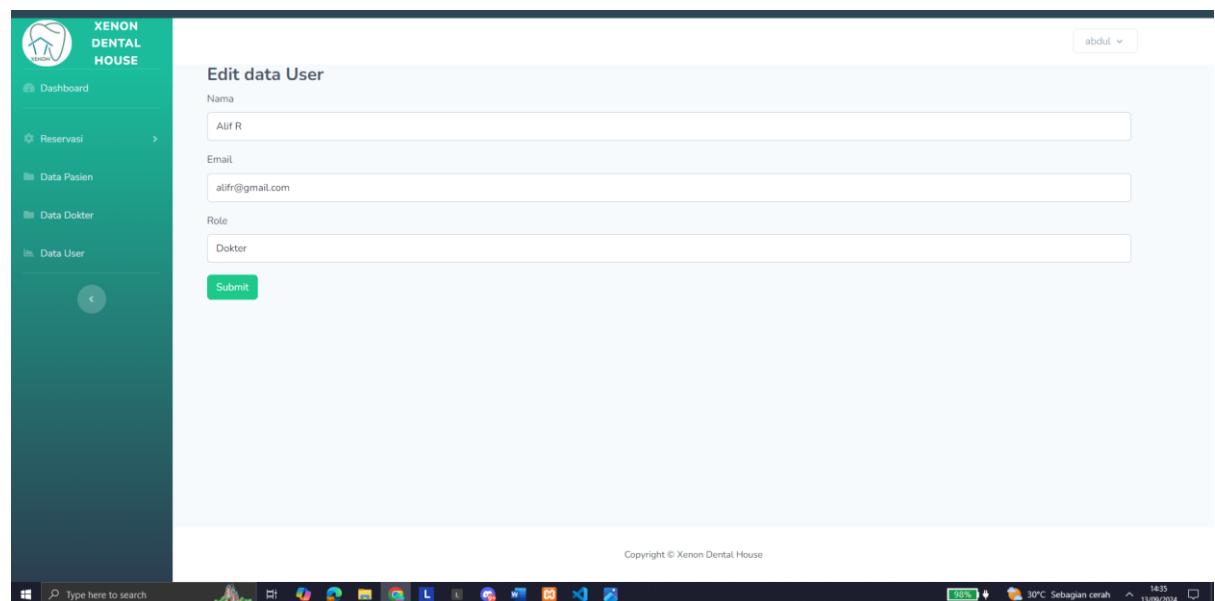


The screenshot shows the 'Tambah data Dokter' (Add Doctor Data) form. The form fields are:

- Nama Dokter (Doctor Name): [Empty input field]
- Alamat (Address): [Empty input field]
- Nomor Hp (Phone Number): [Empty input field]
- Lokasi Praktek (Practice Location): Padang (selected in dropdown)
- Email: [Empty input field]
- Password: [Empty input field]

At the bottom right of the form is a green 'Simpan' (Save) button.

## 18. Implementasi Antarmuka Edit Data User (User Admin)

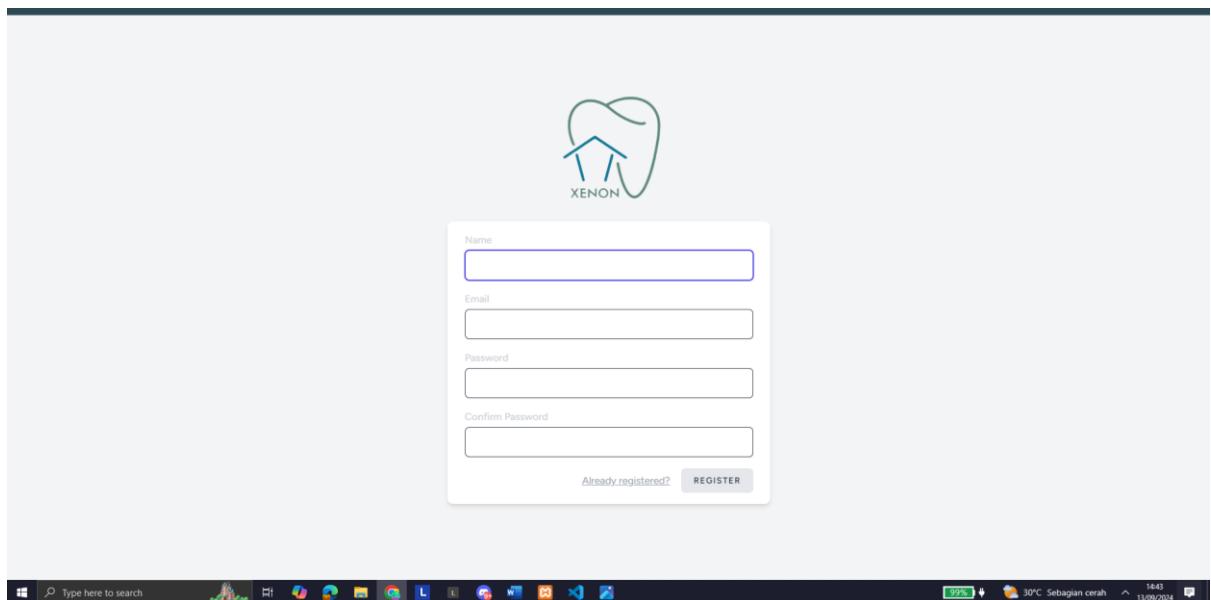


The screenshot shows the 'Edit data User' (Edit User Data) form. The form fields are:

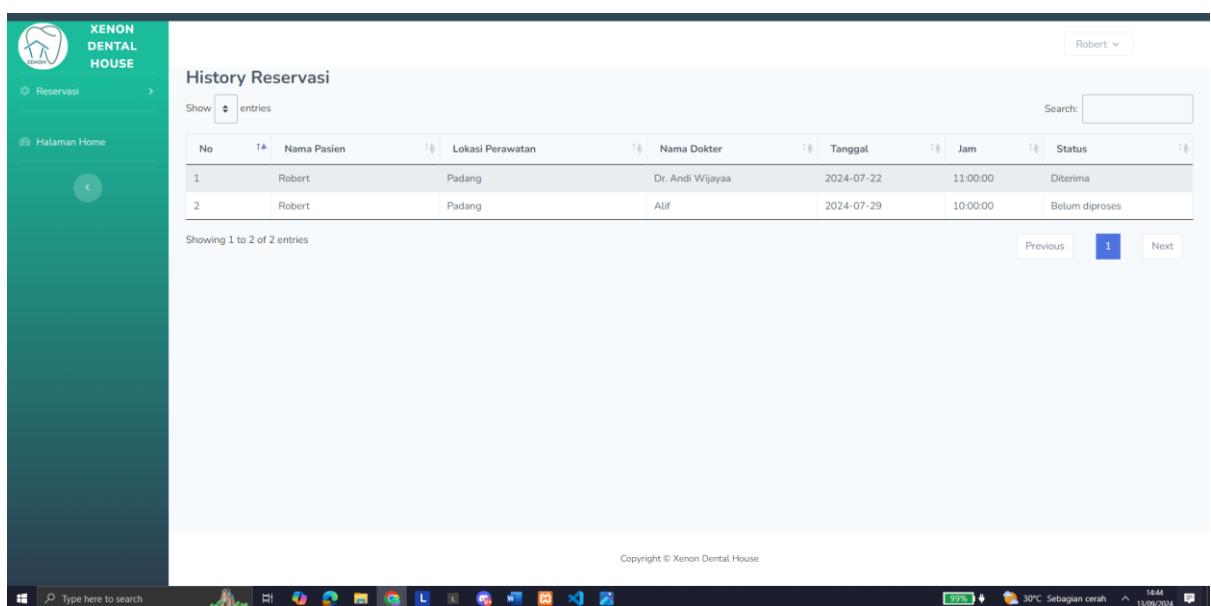
- Nama (Name): Atif R
- Email: alifr@gmail.com
- Role (Role): Dokter

At the bottom right of the form is a green 'Submit' button.

## 19. Implementasi Antarmuka Registrasi Akun (User Pasien)



## 20. Implementasi Antarmuka History Reservasi (User Pasien)



## 21. Implementasi Antarmuka Tambah Reservasi (User Pasien)

The screenshot shows a web-based reservation form titled "Form Tambah Reservasi". The form includes fields for Name, Birthplace, Date of Birth (dd/mm/yyyy), Profession, Address, Phone Number, Treatment (selectable from a dropdown menu), and Location. The left sidebar of the application has a dark teal gradient background and displays navigation links: Reservasi, Halaman Home, Jadwal, Data Pasien, and Data Absen. The top right corner shows a user profile for "Robert". The bottom of the screen shows a Windows taskbar with various pinned icons and system status.

## 22. Implementasi Antarmuka Jadwal Dokter (User Dokter)

The screenshot shows a table titled "Jadwal Saya" (My Schedule) listing two sessions. The columns are labeled "No", "Hari" (Day), and "Sesi" (Session). The data is as follows:

No	Hari	Sesi
1	Senin	1
2	Selasa	2

Below the table, it says "Showing 1 to 2 of 2 entries". The bottom right corner shows page navigation buttons for "Previous", "1", and "Next". The left sidebar and taskbar are identical to the previous screenshot.

## 23. Implementasi Antarmuka Data Pasien (User Dokter)

The screenshot shows a web-based application for managing patient data. The header features the logo 'XENON DENTAL HOUSE' and a user profile 'alif'. The left sidebar includes links for 'Jadwal', 'Data Pasien' (selected), and 'Data Absen'. The main content area is titled 'Data Pasien' and displays a table of patient records. The columns are labeled: no, Tanggal, Jam, Nama Pasien, Lokasi Perawatan, Pekerjaan, Rekam Medik, and Aksi. The data shows four entries:

no	Tanggal	Jam	Nama Pasien	Lokasi Perawatan	Pekerjaan	Rekam Medik	Aksi	
1	2024-10-14	10:00:00	Siti Aminah	Jl. Asia Afrika No.2, Bandung	Guru	082288051901	Rekam Medik	<button>Edit</button> <button>Hapus</button>
2	2024-10-07	13:30:00	Robert	jalan bandung raya	mahasiswa	082288051901	Rekam Medik	<button>Edit</button> <button>Hapus</button>
3	2024-10-07	10:00:00	tono	jalan agung	mahasiswa	082288051901	Rekam Medik	<button>Edit</button> <button>Hapus</button>
4	2024-07-29	10:00:00	dimas	jalan agung	dokter	082288051901	Rekam Medik	<button>Edit</button> <button>Hapus</button>

At the bottom, there is a message 'Showing 1 to 4 of 4 entries' and navigation buttons for 'Previous', 'Next', and page number '1'. The footer includes a copyright notice 'Copyright © Xenon Dental House' and a system status bar with battery level (99%), temperature (30°C), and date/time (14/09/2024).

## 24. Implementasi Antarmuka Hapus Data Pasien (User Dokter)

This screenshot shows the same application interface after a deletion action. A modal dialog box is centered on the screen with the message 'Berhasil!' (Successful!) and 'Data pasien berhasil dihapus!' (Patient data deleted successfully!). It features a large green checkmark icon. A purple 'OK' button is at the bottom right of the dialog. The main data table remains the same as in the previous screenshot, showing six entries. The footer and system status bar are also present.

## 25. Implementasi Antarmuka Edit Data Pasien (User Dokter)

The screenshot shows the 'Edit data Pasien' form. The left sidebar has 'XENON DENTAL HOUSE' at the top, followed by 'Jadwal', 'Data Pasien' (selected), and 'Data Absen'. The main area has fields for Nama Pasien (Siti Aminah), Tempat Lahir (Bandung), Tanggal Lahir (1985-08-24), Pekerjaan (Guru), Alamat (JL Asia Afrika No.2, Bandung), and No HP (082288051901). A blue 'Simpan' button is at the bottom.

## 26. Implementasi Antarmuka Data Perizinan (User Dokter)

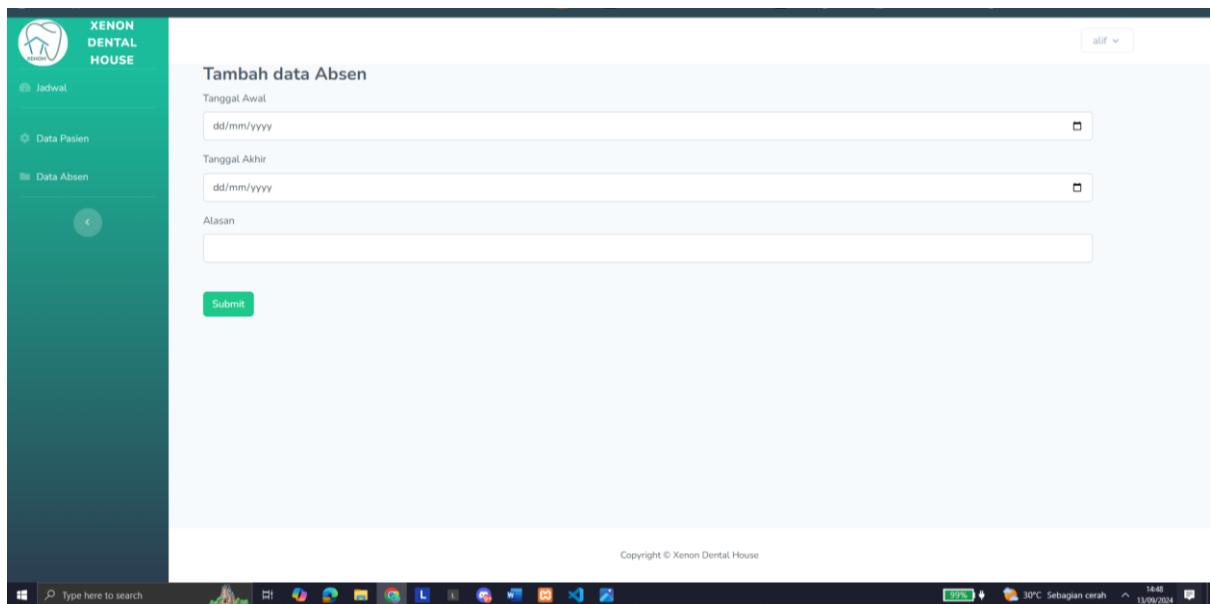
The screenshot shows the 'Data Perizinan' table. The left sidebar has 'Jadwal', 'Data Pasien' (selected), and 'Data Absen'. The table lists 5 entries with columns: No, Tanggal Awal, Tanggal Akhir, Alasan, Status, and Aksi (Edit, Hapus). The entries are:

No	Tanggal Awal	Tanggal Akhir	Alasan	Status	Aksi
1	2024-06-05	2024-06-10	Liburan	Diterima	<button>Edit</button> <button>Hapus</button>
2	2024-06-15	2024-06-20	Pelatihan	Ditolak	<button>Edit</button> <button>Hapus</button>
3	2024-07-04	2024-07-12	berobat sakit	Diterima	<button>Edit</button> <button>Hapus</button>
4	2024-07-29	2024-08-02	kunjungan ke luar kota	Diterima	<button>Edit</button> <button>Hapus</button>
5	2024-07-26	2024-07-27	sakit	Diterima	<button>Edit</button> <button>Hapus</button>

Showing 1 to 5 of 5 entries

[Tambahkan Data Absen](#)

## 27. Implementasi Antarmuka Tambah Data Absen (User Dokter)



Tambah data Absen

Tanggal Awal:

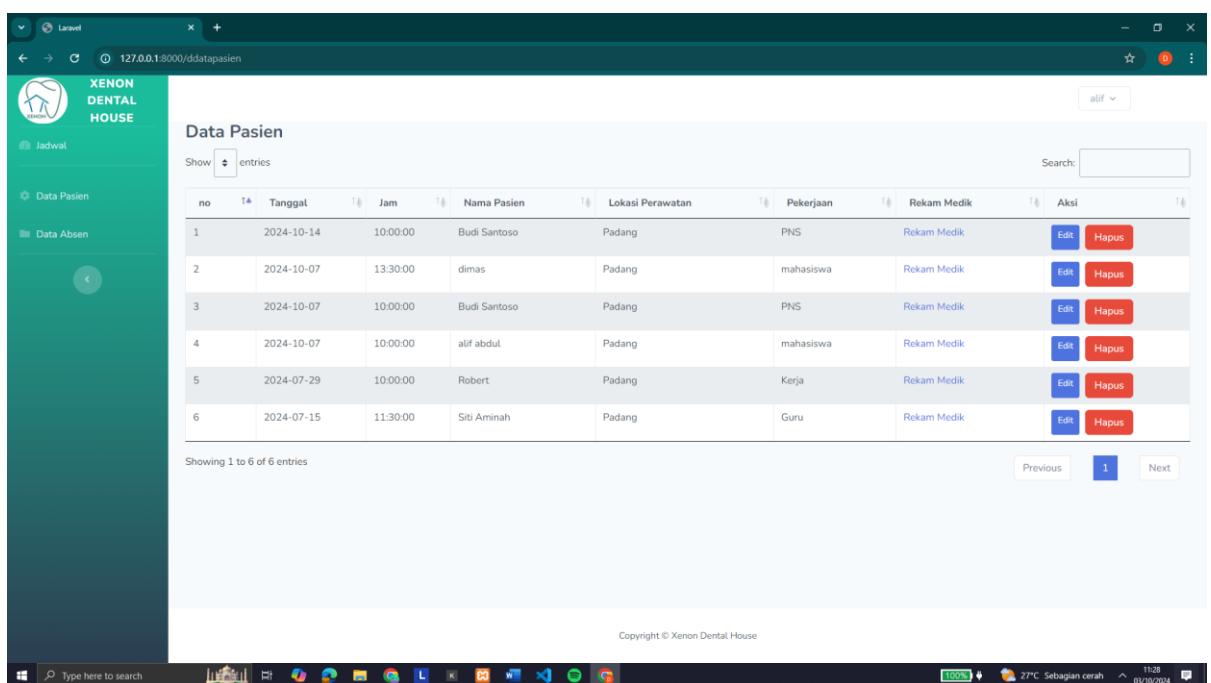
Tanggal Akhir:

Alasan:

Submit

Copyright © Xenon Dental House

## 28. Implementasi Antarmuka Data Pasien (User Dokter)



Data Pasien

no	Tanggal	Jam	Nama Pasien	Lokasi Perawatan	Pekerjaan	Rekam Medik	Aksi
1	2024-10-14	10:00:00	Budi Santoso	Padang	PNS	Rekam Medik	<button>Edit</button> <button>Hapus</button>
2	2024-10-07	13:30:00	dimas	Padang	mahasiswa	Rekam Medik	<button>Edit</button> <button>Hapus</button>
3	2024-10-07	10:00:00	Budi Santoso	Padang	PNS	Rekam Medik	<button>Edit</button> <button>Hapus</button>
4	2024-10-07	10:00:00	alif abdul	Padang	mahasiswa	Rekam Medik	<button>Edit</button> <button>Hapus</button>
5	2024-07-29	10:00:00	Robert	Padang	Kerja	Rekam Medik	<button>Edit</button> <button>Hapus</button>
6	2024-07-15	11:30:00	Siti Aminah	Padang	Guru	Rekam Medik	<button>Edit</button> <button>Hapus</button>

Showing 1 to 6 of 6 entries

Copyright © Xenon Dental House

## 29. Implementasi Antarmuka Edit Data Pasien (User Dokter)

The screenshot shows a web-based application for managing patient data. On the left, there's a sidebar with a logo and navigation links: Dashboard, Reservasi, Data Pasien (selected), Data Dokter, and Data User. The main content area is titled 'Edit data Pasien'. It contains several input fields: 'Nama Pasien' (Siti Aminah), 'Tempat Lahir' (Bandung), 'Tanggal Lahir' (1985-08-24), 'Pekerjaan' (Guru), 'Alamat' (JL Asia Afrika No.2, Bandung), and 'No HP' (082288051901). At the bottom is a blue 'Simpan' button. The footer of the browser window shows the URL '127.0.0.1:8000/ddatapasien/rekam/1' and the text 'Copyright © Xenon Dental House'.

## 30. Implementasi Antarmuka Rekam Medik (User Owner)

The screenshot shows a web-based application for managing medical records. On the left, there's a sidebar with a logo and navigation links: Jadwal, Data Pasien (selected), and Data Absen. The main content area is titled 'Rekam Medik Budi Santoso (2024-10-14)'. It displays a table of medical history items:

Dokter	Alif
Golongan darah	O
Tekanan darah	120 mmHG
Penyakit Jantung	Tidak Ada
Diabetes	Tidak Ada
Hepatitis	Tidak Ada
Penyakit lainnya	Tidak Ada
Alergi Makanan	Tidak Ada
Alergi Obat	Tidak Ada
Keluhan	Sakit pinggang
Perawatan	Behel
Gigi	1
Rencana Tindak Lanjut Behel	besok periksa gusi

At the bottom of the table is a green 'Edit Rekam Medik' button. The footer of the browser window shows the URL '127.0.0.1:8000/ddatapasien/rekam/1' and the text 'Rekam Medik Budi Santoso (2024-10-07)'.

### 31. Implementasi Antarmuka List Perizinan Dokter (User Owner)

Data Perizinan

No	Tanggal Awal	Tanggal Akhir	Alasan	Aksi
1	2024-09-14	2024-09-21	berobat sakit	<button>Terima</button> <button>Tolak</button>

Showing 1 to 1 of 1 entries

Copyright © Xenon Dental House

### 32. Implementasi Antarmuka Data Layanan (User Owner)

Data Layanan

No	Layanan	Harga	Lama Perawatan	Aksi
1	Konsultasi	500000	60	<button>Edit</button> <button>Hapus</button>
2	Behel	300000	120	<button>Edit</button> <button>Hapus</button>
3	Kontrol	100000	30	<button>Edit</button> <button>Hapus</button>
4	Cabut	150000	60	<button>Edit</button> <button>Hapus</button>
5	Bleaching	1000000	60	<button>Edit</button> <button>Hapus</button>

Showing 1 to 5 of 5 entries

[Tambahkan Data Perawatan](#)

Copyright © Xenon Dental House

### 33. Implementasi Antarmuka Tambah Data Layanan (User Owner)

The screenshot shows a software application window titled "Tambah data Layanan". On the left is a green sidebar menu with the "XENON DENTAL HOUSE" logo at the top. Below the logo are several menu items: Dashboard, Data Pasien, Data Dokter, Data Perizinan Dokter, Data Layanan, Data Pemasukan, and Data Pengeluaran. The main content area has three input fields: "Layanan" (Service), "Harga" (Price), and "Estimasi Waktu" (Estimated Time). A green "Simpan" (Save) button is located at the bottom of these fields. The status bar at the bottom right shows "rauf" and the date/time "13/09/2024".

### 34. Implementasi Antarmuka Edit Layanan (User Owner)

The screenshot shows a software application window titled "Edit data Layanan". The layout is identical to the "Tambah data Layanan" screen, featuring a green sidebar menu and a main content area with "Layanan" (Service), "Harga" (Price), and "Estimasi Waktu" (Estimated Time) input fields. The "Harga" field contains the value "500000" and the "Estimasi Waktu" field contains "60". A green "Simpan" (Save) button is at the bottom. The status bar at the bottom right shows "rauf" and the date/time "13/09/2024".

### 35. Implementasi Antarmuka Data Pemasukan (User Owner)

The screenshot shows the 'Data Pemasukan' (Revenue) section of the Xenon Dental House application. The left sidebar includes a logo for 'XENON DENTAL HOUSE' and navigation links for Dashboard, Data Pasien, Data Dokter, Data Perizinan Dokter, Data Layanan, Data Pemasukan (which is highlighted in blue), and Data Pengeluaran. The main content area is titled 'Data Pemasukan' and displays a table of revenue entries. The table columns are: No, Perawatan, Total Pemasukan, Nama Pasien, Nama Dokter, and Tanggal. The data shows three entries: 1. Behel, RP. 300000, Siti Aminah, Alif, 2024-07-15; 2. Behel, RP. 300000, tono, Dr. Andi Wijayaa, 2024-08-05; 3. Behel Bleaching, RP. 320000, dimas, Dr. Andi Wijayaa, 2024-08-01. Below the table, it says 'Showing 1 to 3 of 3 entries'. The bottom right corner shows the current user 'rauf' and a search bar.

No	Perawatan	Total Pemasukan	Nama Pasien	Nama Dokter	Tanggal
1	Behel	RP. 300000	Siti Aminah	Alif	2024-07-15
2	Behel	RP. 300000	tono	Dr. Andi Wijayaa	2024-08-05
3	Behel Bleaching	RP. 320000	dimas	Dr. Andi Wijayaa	2024-08-01

### 36. Implementasi Antarmuka Data Pengeluaran (User Owner)

The screenshot shows the 'Data Pengeluaran' (Expenditure) section of the Xenon Dental House application. The left sidebar is identical to the previous screenshot. The main content area is titled 'Data Pengeluaran' and displays a table of expenditure entries. The table columns are: No, Nama Pengeluaran, Deskripsi Pengeluaran, Jenis Pengeluaran, Jumlah, Tanggal, and Aksi. The data shows three entries: 1. Beli Obat, Pembelian obat pasien, Medis, 200000, 2024-06-01, with 'Edit' and 'Hapus' buttons; 2. Perawatan Alat, Perawatan alat medis, Operasional, 150000, 2024-06-02, with 'Edit' and 'Hapus' buttons; 3. Gaji bulan juli, gaji bulanan semua dokter bulan juli, Gaji, 1000000, 2024-07-16, with 'Edit' and 'Hapus' buttons. Below the table, it says 'Showing 1 to 3 of 3 entries'. At the bottom left is a green button labeled 'Tambahkan Data Pengeluaran'. The bottom right corner shows the current user 'rauf' and a search bar.

No	Nama Pengeluaran	Deskripsi Pengeluaran	Jenis Pengeluaran	Jumlah	Tanggal	Aksi
1	Beli Obat	Pembelian obat pasien	Medis	200000	2024-06-01	<button>Edit</button> <button>Hapus</button>
2	Perawatan Alat	Perawatan alat medis	Operasional	150000	2024-06-02	<button>Edit</button> <button>Hapus</button>
3	Gaji bulan juli	gaji bulanan semua dokter bulan juli	Gaji	1000000	2024-07-16	<button>Edit</button> <button>Hapus</button>

### 37. Implementasi Antarmuka Tambah Data Pengeluaran (User Owner)

The screenshot shows the Xenon Dental House software interface. On the left is a green sidebar menu with the logo 'XENON DENTAL HOUSE'. The main area displays a form titled 'Tambah data Pengeluaran' (Add Expense Data). The form contains the following fields:

- Nama Pengeluaran (Expense Name)
- Deskripsi Pengeluaran (Expense Description)
- Jenis Pengeluaran (Expense Type)
- Jumlah Pengeluaran (Expense Amount)
- Tanggal (Date) with a dd/mm/yyyy input field

A blue 'Simpan' (Save) button is located at the bottom of the form. The status bar at the bottom right shows 'rauf' and system information like battery level (89%), temperature (30°C), and date (13/09/2024).

### 38. Implementasi Antarmuka Edit Data Pengeluaran (User Owner)

The screenshot shows the Xenon Dental House software interface. On the left is a green sidebar menu with the logo 'XENON DENTAL HOUSE'. The main area displays a form titled 'Edit data Pengeluaran' (Edit Expense Data). The form contains the following fields, showing sample data:

- Nama Pengeluaran (Expense Name): Beli Obat
- Deskripsi Pengeluaran (Expense Description): Pembelian obat pasien
- Jenis Pengeluaran (Expense Type): Medis
- Jumlah Pengeluaran (Expense Amount): 200000
- Tanggal (Date): 01/06/2024

A blue 'Simpan' (Save) button is located at the bottom of the form. The status bar at the bottom right shows 'rauf' and system information like battery level (89%), temperature (30°C), and date (13/09/2024).

**LAMPIRAN E**  
**KODE PROGRAM**  
**(LANJUTAN)**

## Kode Route

### 1. Auth.php

```
<?php

use App\Http\Controllers\Auth\AuthenticatedSessionController;
use App\Http\Controllers\Auth\ConfirmablePasswordController;
use App\Http\Controllers\Auth\EmailVerificationNotificationController;
use App\Http\Controllers\Auth\EmailVerificationPromptController;
use App\Http\Controllers\Auth\NewPasswordController;
use App\Http\Controllers\Auth\PasswordController;
use App\Http\Controllers\Auth\PasswordResetLinkController;
use App\Http\Controllers\Auth\RegisteredUserController;
use App\Http\Controllers\Auth\VerifyEmailController;
use Illuminate\Support\Facades\Route;

Route::middleware('guest')->group(function () {
    Route::get('register', [RegisteredUserController::class, 'create'])
        ->name('register');

    Route::post('register', [RegisteredUserController::class, 'store']);

    Route::get('login', [AuthenticatedSessionController::class, 'create'])
        ->name('login');

    Route::post('login', [AuthenticatedSessionController::class, 'store']);

    Route::get('forgot-password', [PasswordResetLinkController::class, 'create'])
        ->name('password.request');

    Route::post('forgot-password', [PasswordResetLinkController::class, 'store'])
        ->name('password.email');

    Route::get('reset-password/{token}', [NewPasswordController::class, 'create'])
        ->name('password.reset');

    Route::post('reset-password', [NewPasswordController::class, 'store'])
        ->name('password.store');
});

Route::middleware('auth')->group(function () {
    Route::get('verify-email', EmailVerificationPromptController::class)
        ->name('verification.notice');

    Route::get('verify-email/{id}/{hash}', VerifyEmailController::class)
        ->middleware(['signed', 'throttle:6,1'])
        ->name('verification.verify');

    Route::post('email/verification-notification',
        [EmailVerificationNotificationController::class, 'store'])
        ->middleware('throttle:6,1')
        ->name('verification.send');

    Route::get('confirm-password', [ConfirmablePasswordController::class, 'show'])
        ->name('password.confirm');

    Route::post('confirm-password', [ConfirmablePasswordController::class, 'store']);

    Route::put('password', [PasswordController::class, 'update'])->name('password.update');

    Route::post('logout', [AuthenticatedSessionController::class, 'destroy'])
        ->name('logout');
});
```

### 2. Web.php

```
<?php

use Illuminate\Support\Facades\Route;
use App\Http\Controllers\ProfileController;
use App\Http\Controllers\AbsenceController;
use App\Http\Controllers\Admin\DataDokterController;
use App\Http\Controllers\Admin\HomeController;
use App\Http\Controllers\Dokter\IzinController;
```

```

use App\Http\Controllers\Admin\PasienController;
use App\Http\Controllers\Admin\UserController;
use App\Http\Controllers\Admin\JadwalController;
use App\Http\Controllers\Admin\RekamController;
use App\Http\Controllers\Admin\ReservasiController;
use App\Http\Controllers\Dokter\DokterHomeController;
use App\Http\Controllers\Owner\PerawatanController;
use App\Http\Controllers\Owner\PengeluaranController;
use App\Http\Controllers\Owner\PemasukanController;
use App\Http\Controllers\Pasien\ReservasiPasienController;
use App\Http\Controllers\WhatsAppController;

use App\Models\Perawatan;

require __DIR__.'/auth.php';

Route::get('/welcome', function () {
    return view('welcome');
});

Route::get('/manifest.json', function () {
    return response()->file(public_path('manifest.json'));
});

Route::get('/serviceworker.js', function () {
    return response()->file(public_path('serviceworker.js'));
});

Route::post('/send-whatsapp', [WhatsAppController::class, 'sendWhatsAppMessage'])->name('whatsapp.send');

Route::middleware('auth')->group(function () {
    Route::get('/home', function () {
        return view('landing_page');
    });
    Route::get('/pasien/history', [ReservasiPasienController::class, 'index'])->name('pasien.history');
    Route::get('/pasien/add', [ReservasiController::class, 'add'])->name('pasien.add');
    Route::post('/pasien/insert', [ReservasiPasienController::class, 'insert'])->name('pasien.insert');
    Route::get('/api/dokter-by-lokasi-p', [ReservasiPasienController::class, 'getDokterByLokasi']);
    Route::get('/api/unavailable-times-p', [ReservasiPasienController::class, 'getUnavailableTimes']);
    Route::get('/api/available-dates-p', [ReservasiPasienController::class, 'getAvailableDates']);
    Route::get('/api/available-days-p', [ReservasiPasienController::class, 'getAvailableDays']);
    Route::get('/api/booked-times-p', [ReservasiPasienController::class, 'getBookedTimes']);
    Route::get('/api/sesi-by-dokter-and-hari-p', [ReservasiPasienController::class, 'getSesiByDokterAndHari']);

    Route::get('/profile', [ProfileController::class, 'edit'])->name('profile.edit');
    Route::patch('/profile', [ProfileController::class, 'update'])->name('profile.update');
    Route::delete('/profile', [ProfileController::class, 'destroy'])->name('profile.destroy');
});

Route::get('/', function () {
    return view('landing_page');
});
Route::get('/offline', function () {
    return view('offline');
});

Route::middleware(['auth', 'verified'])->group(function () {
    Route::get('/admin', [HomeController::class, 'index'])->name('admin');
    Route::post('/admin/filter', [HomeController::class, 'filter'])->name('home.filter');
});

Route::middleware(['role:admin'])->group(function () {
    // Routes untuk DataDokter
    Route::get('/datadokter', [DataDokterController::class, 'index'])->name('datadokter.index');
    Route::get('/datadokter/add', [DataDokterController::class, 'add'])->name('datadokter.add');
});

```

```

        Route::post('/datadokter/insert', [DataDokterController::class, 'insert'])-
>name('datadokter.insert');
        Route::get('/datadokter/edit/{id}', [DataDokterController::class, 'edit'])-
>name('datadokter.edit');
        Route::post('/datadokter/update/{id}', [DataDokterController::class, 'update'])-
>name('datadokter.update');
        Route::delete('/datadokter/destroy/{id}', [DataDokterController::class, 'destroy'])-
>name('datadokter.destroy');

        // Routes untuk Jadwal DataDokter
        Route::get('/datadokter/jadwal/{id}', [JadwalController::class, 'index'])-
>name('dokterjadwal.index');
        Route::get('/datadokter/jadwal/add/{id}', [JadwalController::class, 'add'])-
>name('dokterjadwal.add');
        Route::post('/datadokter/jadwal/insert/{id}', [JadwalController::class, 'insert'])-
>name('dokterjadwal.insert');
        Route::get('/datadokter/jadwal/edit/{id}', [JadwalController::class, 'edit'])-
>name('dokterjadwal.edit');
        Route::post('/datadokter/jadwal/update/{id}', [JadwalController::class, 'update'])-
>name('dokterjadwal.update');
        Route::delete('/datadokter/jadwal/destroy/{id}', [JadwalController::class, 'destroy'])-
>name('dokterjadwal.destroy');

        // Routes untuk Reservasi Admin
        Route::get('/reservasi/admin/', [ReservasiController::class, 'index'])-
>name('reservasi.index');
        Route::get('/reservasi/admin/add/', [ReservasiController::class, 'add'])-
>name('reservasi.add');
        Route::post('/reservasi/admin/insert/', [ReservasiController::class, 'insert'])-
>name('reservasi.insert');
        Route::get('/reservasi/admin/edit/{id}', [ReservasiController::class, 'edit'])-
>name('reservasi.edit');
        Route::post('/reservasi/admin/update/{id}', [ReservasiController::class, 'update'])-
>name('reservasi.update');
        Route::delete('/reservasi/admin/destroy/{id}', [ReservasiController::class, 'destroy'])-
>name('reservasi.destroy');
        Route::post('/reservasi/admin/terima/{id}', [ReservasiController::class, 'terima'])-
>name('reservasi.terima');
        Route::post('/reservasi/admin/tolak/{id}', [ReservasiController::class, 'tolak'])-
>name('reservasi.tolak');
        Route::get('/reservasi/pasien/', [ReservasiController::class, 'pasien'])-
>name('reservasi.pasien');
        Route::get('/api/dokter-by-lokasi', [ReservasiController::class, 'getDokterByLokasi']);
        Route::get('/api/unavailable-times', [ReservasiController::class, 'getUnavailableTimes']);
        Route::get('/api/available-dates', [ReservasiController::class, 'getAvailableDates']);
        Route::get('/api/available-days', [ReservasiController::class, 'getAvailableDays']);
        Route::get('/api/booked-times', [ReservasiController::class, 'getBookedTimes']);

        Route::get('/api/seji-by-dokter-and-hari', [ReservasiController::class,
'getSesiByDokterAndHari']);
        Route::get('/reservasi/getDoctors', [ReservasiController::class, 'getDokterByLokasi'])-
>name('reservasi.getDoctors');
        Route::get('/reservasi/getTimeSlots', [ReservasiController::class, 'getTimeSlots'])-
>name('reservasi.getTimeSlots');

        // Routes untuk DataPasien
        Route::get('/adatapasien', [PasienController::class, 'index'])->name('datapasien.index');
        Route::get('/adatapasien/add', [PasienController::class, 'add'])->name('datapasien.add');
        Route::post('/adatapasien/insert', [PasienController::class, 'insert'])-
>name('datapasien.insert');
        Route::get('/adatapasien/edit/{id}', [PasienController::class, 'edit'])-
>name('datapasien.edit');
        Route::post('/adatapasien/update/{id}', [PasienController::class, 'update'])-
>name('datapasien.update');
        Route::delete('/adatapasien/destroy/{id}', [PasienController::class, 'destroy'])-
>name('datapasien.destroy');

        // Routes untuk Rekam Pasien
        Route::get('/datapasien/rekam/{id}', [RekamController::class, 'index'])-
>name('admin.rekampasien');
        Route::get('/datapasien/rekam/edit/{id}', [RekamController::class, 'edit'])-
>name('admin.rekampasien.edit');
        Route::post('/datapasien/rekam/update/{id}', [RekamController::class, 'update'])-
>name('admin.rekampasien.update');

        // Routes untuk DataUser
        Route::get('/datauser', [UserController::class, 'index'])->name('datauser.index');
        Route::get('/datauser/add', [UserController::class, 'add'])->name('datauser.add');

```

```

        Route::post('/datauser/insert', [UserController::class, 'insert'])-
>name('datauser.insert');
        Route::get('/datauser/edit/{id}', [UserController::class, 'edit'])->name('datauser.edit');
        Route::post('/datauser/update/{id}', [UserController::class, 'update'])-
>name('datauser.update');
        Route::delete('/datauser/destroy/{id}', [UserController::class, 'destroy'])-
>name('datauser.destroy');

        Route::get('/datauserdokter/add', [UserController::class, 'add_dokter'])-
>name('datauserdokter.add');
        Route::post('/datauserdokter/insert', [UserController::class, 'insert_dokter'])-
>name('datauserdokter.insert');
    });

Route::get('/profil', function () {
    return view('admin.profil');
});

// Routes untuk dokter

Route::middleware(['role:Dokter'])->group(function () {
    Route::get('/dokter',[DokterHomeController::class, 'index'])->name('dokter.index');
    Route::get('/dataabsen',[IzinController::class, 'index'])->name('dataabsen.index');
    Route::get('/dataabsen/add',[IzinController::class, 'add'])->name('dataabsen.add');
    Route::get('/dataabsen/edit{id}',[IzinController::class, 'edit'])->name('dataabsen.edit');
    Route::post('/dataabsen/update{id}',[IzinController::class, 'update'])-
>name('dataabsen.update');
    Route::post('/dataabsen/insert',[IzinController::class, 'insert'])-
>name('dataabsen.insert');
    Route::delete('/dataabsen/destroy{id}',[IzinController::class, 'destroy'])-
>name('dataabsen.destroy');

    Route::get('/ddatapasien',[PasienController::class, 'index'])->name('ddatapasien.index');
    Route::get('/ddatapasien/add',[PasienController::class, 'add'])->name('ddatapasien.add');
    Route::post('/ddatapasien/insert',[PasienController::class, 'inser'])-
>name('ddatapasien.insert');
    Route::get('/ddatapasien/edit{id}',[PasienController::class, 'edit'])-
>name('ddatapasien.edit');
    Route::post('/ddatapasien/update{id}',[PasienController::class, 'update'])-
>name('ddatapasien.update');
    Route::delete('/ddatapasien/destroy{id}',[PasienController::class, 'destroy'])-
>name('ddatapasien.destroy');
    Route::get('/ddatapasien/rekam/{id}',[RekamController::class, 'index'])-
>name('dokter.rekampasien');
    Route::get('/ddatapasien/rekam/edit{id}',[RekamController::class, 'edit'])-
>name('dokter.rekampasien.edit');
    Route::post('/ddatapasien/rekam/update{id}',[RekamController::class, 'update'])-
>name('dokter.rekampasien.update');
});

Route::get('/kalender', function () {
    return view('absence');
});

// Route::resource('absences', AbsenceController::class);

Route::middleware(['role:owner'])->group(function () {
    // Routes untuk owner
    // Route::get('/owner', function () {
    //     return view('owner.home');
    // });
    Route::get('/owner', [HomeController::class, 'index'])->name('owner.index');
    Route::post('/owner/filter', [HomeController::class, 'filter'])->name('owner.filter');

    Route::get('/odatadokter', [DataDokterController::class, 'index'])-
>name('odatadokter.index');

    // Routes untuk Jadwal DataDokter
    Route::get('/odatadokter/jadwal/{id}', [JadwalController::class, 'index'])-
>name('odatadokter.jadwal');

    Route::get('/odatapasien', [PasienController::class, 'index'])->name('odatapasien.index');
    Route::get('/datalayanan', [PerawatanController::class, 'index'])->name('layanan.index');
    Route::get('/datalayanan/add',[PerawatanController::class, 'add'])->name('layanan.add');
    Route::post('/datalayanan/insert',[PerawatanController::class, 'insert'])-
>name('layanan.insert');
    Route::get('/datalayanan/edit{id}',[PerawatanController::class, 'edit'])-
>name('layanan.edit');
});

```

```

        Route::post('/datalayanan/update/{id}', [PerawatanController::class, 'update'])-
>name('layanan.update');
        Route::delete('/datalayanan/destroy/{id}', [PerawatanController::class, 'destroy'])-
>name('layanan.destroy');

        Route::get('/datapengeluaran', [PengeluaranController::class, 'index'])-
>name('pengeluaran.index');
        Route::get('/datapengeluaran/add', [PengeluaranController::class, 'add'])-
>name('pengeluaran.add');
        Route::post('/datapengeluaran/insert', [PengeluaranController::class, 'insert'])-
>name('pengeluaran.insert');
        Route::get('/datapengeluaran/edit/{id}', [PengeluaranController::class, 'edit'])-
>name('pengeluaran.edit');
        Route::post('/datapengeluaran/update/{id}', [PengeluaranController::class, 'update'])-
>name('pengeluaran.update');
        Route::delete('/datapengeluaran/destroy/{id}', [PengeluaranController::class, 'destroy'])-
>name('pengeluaran.destroy');

        Route::get('/datapemasukan', [PemasukanController::class, 'index'])-
>name('pemasukan.index');

        Route::get('/pengeluaran/edit', function () {
            return view('owner.datapengeluaran_edit');
        });
        Route::get('/pengeluaran/add', function () {
            return view('owner.datapengeluaran_add');
        });
        Route::get('/odatapasien/rekam/{id}', [RekamController::class, 'index'])-
>name('owner.rekampasien');
        Route::get('/owner/izin',[IzinController::class, 'index'])->name('owner.izin.index');
        Route::post('/owner/izin/terima/{id}',[IzinController::class, 'terima'])-
>name('owner.izin.terima');
        Route::post('/owner/izin/tolak/{id}',[IzinController::class, 'tolak'])-
>name('owner.izin.tolak');

    });

```

## Kode Controller Admin

### 1. DataDokterController.php

```

<?php

namespace App\Http\Controllers\Admin;

use Session;
use App\Models\Lokasi;
use App\Models\DataDokter;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use App\Http\Controllers\Controller;
use Illuminate\Support\Facades\Auth;
use RealRashid\SweetAlert\Facades\Alert;

```

```

class DataDokterController extends Controller
{
    protected $datadokter;

    public function __construct(DataDokter $datadokter)
    {
        $this->datadokter = $datadokter;
    }

    public function index()
    {
        $datadokter = [
            'datadokter' => $this->datadokter->allData(),
        ];
        $user = Auth::user();
        if ($user->role === 'owner') {
            return view('owner.datadokter', $datadokter);
        }
    }
}

```

```

        else{
            return view('admin.datadokter', $datadokter);
        }
    }

    public function add()
    {
        $lokasi = Lokasi::all();
        return view('admin.datadokter_add', compact('lokasi'));
    }

    public function insert(Request $request)
    {
        $request->validate([
            'nama' => 'required',
            'alamat' => 'required',
            'nomor_hp' => 'required',
            'lokasi_id'=> 'required',
        ], [
            'nama.required' => 'Nama harus diisi!',
            'alamat.required' => 'alamat harus diisi!',
            'nomor_hp.required' => 'nomor_hp harus diisi!',
            'lokasi_id.required' => 'lokasi praktek harus diisi'
        ]);
    }

    $data = [
        'nama' => $request->nama,
        'alamat' => $request->alamat,
        'nomor_hp' => $request->nomor_hp,
        'lokasi_id' => $request->lokasi_id
    ];

    $this->datadokter->addData($data);
    Alert::success('Berhasil!', 'Data dokter berhasil ditambahkan!');
    return redirect('/datadokter');
}

public function destroy($dokter_id)
{
    DB::table('dokter')->where('dokter_id', $dokter_id)->delete();

    // Menampilkan alert sukses dan mengarahkan kembali ke halaman data dokter
    Alert::success('Berhasil!', 'Data dokter berhasil dihapus!');
    return redirect('/datadokter');
}

public function edit($id)
{
    $datadokter = DB::table('dokter')->where('dokter_id', $id)->first();
    return view('admin.datadokter_edit', ['datadokter' => $datadokter]);
}

public function detail($id)
{
    $datajadwal = DB::table('detail_jadwal')->where('dokter_id', $id)->get();
    return view('admin.datadokterjadwal', ['datajadwal' => $datajadwal]);
}

public function update(Request $request, $id)
{
    $request->validate([
        'nama' => 'required',
        'alamat' => 'required',
        'nomor_hp' => 'required',
    ]);

    $data = [
        'nama' => $request->nama,
        'alamat' => $request->alamat,
        'nomor_hp' => $request->nomor_hp,
    ];

    DB::table('dokter')->where('dokter_id', $id)->update($data);
    Alert::success('Berhasil!', 'Data dokter berhasil diupdate!');
    return redirect('/datadokter');
}
}

```

## 2. HomeController.php

```
<?php

namespace App\Http\Controllers\Admin;

use App\Http\Controllers\Controller;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use Carbon\Carbon;
use Illuminate\Support\Facades\Auth;
use App\Models\Izin;

class HomeController extends Controller
{
    public function index()
    {

        $today = Carbon::today()->toDateString();
        $absences = Izin::where('tanggal_awal', '<=', $today)
                        ->where('tanggal_akhir', '>=', $today)
                        ->get();

        $currentMonth = Carbon::now()->format('Y-m');
        $visitsToday = DB::table('reservasi_rekam_medik')
                        ->whereDate('tanggal', Carbon::today())
                        ->count();
        $visitsThisMonth = $this->getMonthlyVisits($currentMonth);

        $visitStats = [
            'today' => $visitsToday,
            'week1' => $visitsThisMonth['week1'],
            'week2' => $visitsThisMonth['week2'],
            'week3' => $visitsThisMonth['week3'],
            'week4' => $visitsThisMonth['week4'],
            'this_month' => $visitsThisMonth['this_month'],
        ];
        $visits = DB::table('reservasi_rekam_medik')
                    ->whereYear('tanggal', Carbon::now()->year)
                    ->whereMonth('tanggal', Carbon::now()->month)
                    ->count();
        $doctor = DB::table("Dokter")
                    ->count();

        $location = DB::table("lokasi")
                    ->count();

        // Calculate patient stats
        $patientStats = DB::table('Pasien')
                        ->join('reservasi_rekam_medik', 'Pasien.pasien_id', '=',
                            'reservasi_rekam_medik.pasien_id')
                        ->whereYear('reservasi_rekam_medik.tanggal', Carbon::now()->year)
                        ->whereMonth('reservasi_rekam_medik.tanggal', Carbon::now()->month)
                        ->select(
                            DB::raw("SUM(CASE WHEN DATEDIFF(CURDATE(), tanggal_lahir) / 365.25 < 2 THEN 1 ELSE
0 END) as bayi"),
                            DB::raw("SUM(CASE WHEN DATEDIFF(CURDATE(), tanggal_lahir) / 365.25 BETWEEN 2 AND
12 THEN 1 ELSE 0 END) as anak"),
                            DB::raw("SUM(CASE WHEN DATEDIFF(CURDATE(), tanggal_lahir) / 365.25 BETWEEN 13 AND
19 THEN 1 ELSE 0 END) as remaja"),
                            DB::raw("SUM(CASE WHEN DATEDIFF(CURDATE(), tanggal_lahir) / 365.25 BETWEEN 20 AND
59 THEN 1 ELSE 0 END) as dewasa"),
                            DB::raw("SUM(CASE WHEN DATEDIFF(CURDATE(), tanggal_lahir) / 365.25 >= 60 THEN 1
ELSE 0 END) as lansia")
                        )->first();

        // Fetch treatments for the current month
        $treatments = DB::table('Perawatan')
                        ->join('perawatan_reservasi', 'Perawatan.perawatan_id', '=',
                            'perawatan_reservasi.perawatan_id')
                        ->join('reservasi_rekam_medik', 'perawatan_reservasi.reservasi_id', '=',
                            'reservasi_rekam_medik.reservasi_id')
                        ->whereYear('reservasi_rekam_medik.tanggal', Carbon::now()->year)
                        ->whereMonth('reservasi_rekam_medik.tanggal', Carbon::now()->month)
                        ->select('Perawatan.jenis_Perawatan', DB::raw('count(*) as total'))
                        ->groupBy('Perawatan.jenis_Perawatan')
                        ->get();

        // Fetch doctor performance for the current month
        $doctorPerformance = DB::table('Dokter')
```

```

->join('reservasi_rekam_medik', 'Dokter.dokter_id', '=',
'reservasi_rekam_medik.dokter_id')
->whereYear('reservasi_rekam_medik.tanggal', Carbon::now()->year)
->whereMonth('reservasi_rekam_medik.tanggal', Carbon::now()->month)
->select('Dokter.nama', DB::raw('count(*) as total'))
->groupBy('Dokter.nama')
->get();

$totalIncome = DB::table('Perawatan')
->join('perawatan_reservasi', 'Perawatan.perawatan_id', '=',
'perawatan_reservasi.perawatan_id')
->join('reservasi_rekam_medik', 'perawatan_reservasi.reservasi_id', '=',
'reservasi_rekam_medik.reservasi_id')
->where('reservasi_rekam_medik.draft', 0)
->whereYear('reservasi_rekam_medik.tanggal', Carbon::now()->year)
->whereMonth('reservasi_rekam_medik.tanggal', Carbon::now()->month)
->sum('perawatan_reservasi.harga');

$totalExpenses = DB::table('pengeluaran')
->whereYear('tanggal', Carbon::now()->year)
->whereMonth('tanggal', Carbon::now()->month)
->sum('jumlah_pengeluaran');

// Calculate profit
$profit = $totalIncome - $totalExpenses;

// Determine the user role and return the appropriate view
$user = Auth::user();
if ($user->role === 'owner') {
    return view('owner.home', compact('visitStats', 'currentMonth', 'visits', 'doctor',
'location', 'treatments', 'patientStats', 'doctorPerformance', 'profit'));
}

else{
    return view('admin.home', compact('absences', 'visitStats', 'currentMonth',
'vests', 'doctor', 'location', 'treatments', 'patientStats', 'doctorPerformance', 'profit'));
}

public function filter(Request $request)
{
    $currentMonth = Carbon::now()->format('Y-m');
    $month = $request->input('month', $currentMonth);

    $today = Carbon::today()->toDateString();
    $absences = Izin::where('tanggal_awal', '<=', $today)
->where('tanggal_akhir', '>=', $today)
->get();
    $year = substr($month, 0, 4);
    $monthNumber = substr($month, 5, 2);

    $visits = DB::table('reservasi_rekam_medik')
->whereYear('tanggal', $year)
->whereMonth('tanggal', $monthNumber)
->count();
    $doctor = DB::table("Dokter")->count();
    $location = DB::table("lokasi")->count();

    $visitsToday = DB::table('reservasi_rekam_medik')
->whereDate('tanggal', Carbon::today())
->count();

    $visitsInMonth = DB::table('reservasi_rekam_medik')
->whereYear('tanggal', $year)
->whereMonth('tanggal', $monthNumber)
->get();

    $week1 = $visitsInMonth->filter(function ($visit) {
        return Carbon::parse($visit->tanggal)->weekOfMonth === 1;
    })->count();

    $week2 = $visitsInMonth->filter(function ($visit) {
        return Carbon::parse($visit->tanggal)->weekOfMonth === 2;
    })->count();

    $week3 = $visitsInMonth->filter(function ($visit) {
        return Carbon::parse($visit->tanggal)->weekOfMonth === 3;
    })->count();
}

```

```

$week4 = $visitsInMonth->filter(function ($visit) {
    return Carbon::parse($visit->tanggal)->weekOfMonth === 4;
})->count();

$visitStats = [
    'today' => $visitsToday,
    'week1' => $week1,
    'week2' => $week2,
    'week3' => $week3,
    'week4' => $week4,
    'this_month' => $week1+$week2+$week3+$week4,
];

$totalIncome = DB::table('Perawatan')
    ->join('perawatan_reservasi', 'Perawatan.perawatan_id', '=',
'reparawatan_reservasi.perawatan_id')
    ->join('reservasi_rekam_medik', 'perawatan_reservasi.reservasi_id', '=',
'reservasi_rekam_medik.reservasi_id')
    ->whereYear('reservasi_rekam_medik.tanggal', $year)
    ->whereMonth('reservasi_rekam_medik.tanggal', $monthNumber)
    ->sum('perawatan_reservasi.harga');

// Calculate total expenses for the selected month
$totalExpenses = DB::table('pengeluaran')
    ->whereYear('tanggal', $year)
    ->whereMonth('tanggal', $monthNumber)
    ->sum('jumlah_pengeluaran');

// Calculate profit
$profit = $totalIncome - $totalExpenses;

// Fetch treatments for the selected month
$treatments = DB::table('Perawatan')
    ->join('perawatan_reservasi', 'Perawatan.perawatan_id', '=',
'reparawatan_reservasi.perawatan_id')
    ->join('reservasi_rekam_medik', 'perawatan_reservasi.reservasi_id', '=',
'reservasi_rekam_medik.reservasi_id')
    ->whereYear('reservasi_rekam_medik.tanggal', $year)
    ->whereMonth('reservasi_rekam_medik.tanggal', $monthNumber)
    ->select('Perawatan.jenis_Perawatan', DB::raw('count(*) as total'))
    ->groupBy('Perawatan.jenis_Perawatan')
    ->get();

// Calculate patient stats for the selected month
$patientStats = DB::table('Pasien')

    ->join('reservasi_rekam_medik', 'Pasien.pasien_id', '=',
'reservasi_rekam_medik.pasien_id')
    ->whereYear('reservasi_rekam_medik.tanggal', $year)
    ->whereMonth('reservasi_rekam_medik.tanggal', $monthNumber)
    ->select(
        DB::raw("SUM(CASE WHEN DATEDIFF(CURDATE(), tanggal_lahir) / 365.25 < 2 THEN 1 ELSE
0 END) as bayi"),
        DB::raw("SUM(CASE WHEN DATEDIFF(CURDATE(), tanggal_lahir) / 365.25 BETWEEN 2 AND
12 THEN 1 ELSE 0 END) as anak"),
        DB::raw("SUM(CASE WHEN DATEDIFF(CURDATE(), tanggal_lahir) / 365.25 BETWEEN 13 AND
19 THEN 1 ELSE 0 END) as remaja"),
        DB::raw("SUM(CASE WHEN DATEDIFF(CURDATE(), tanggal_lahir) / 365.25 BETWEEN 20 AND
59 THEN 1 ELSE 0 END) as dewasa"),
        DB::raw("SUM(CASE WHEN DATEDIFF(CURDATE(), tanggal_lahir) / 365.25 >= 60 THEN 1
ELSE 0 END) as lansia")
    )->first();

// Fetch doctor performance for the selected month
$doctorPerformance = DB::table('Dokter')
    ->join('reservasi_rekam_medik', 'Dokter.dokter_id', '=',
'reservasi_rekam_medik.dokter_id')
    ->whereYear('reservasi_rekam_medik.tanggal', $year)
    ->whereMonth('reservasi_rekam_medik.tanggal', $monthNumber)
    ->select('Dokter.nama', DB::raw('count(*) as total'))
    ->groupBy('Dokter.nama')
    ->get();

// Determine the user role and return the appropriate view
$user = Auth::user();
if ($user->role === 'owner') {
    return view('owner.home', compact('absences', 'visitStats', 'currentMonth',
'month', 'visits', 'doctor', 'location', 'treatments', 'patientStats', 'doctorPerformance',
'profit'));
} else {
}

```

```

        return view('admin.home', compact('absences', 'visitStats', 'currentMonth',
'month', 'visits', 'doctor', 'location', 'treatments', 'patientStats', 'doctorPerformance',
'profit'));
    }
}

private function getMonthlyVisits($month)
{
    $year = substr($month, 0, 4);
    $monthNumber = substr($month, 5, 2);

    $visits = DB::table('reservasi_rekam_medik')
->whereYear('tanggal', $year)
->whereMonth('tanggal', $monthNumber)
->get();

$week1 = $visits->filter(function ($visit) {
    return Carbon::parse($visit->tanggal)->weekOfMonth === 1;
})->count();

$week2 = $visits->filter(function ($visit) {
    return Carbon::parse($visit->tanggal)->weekOfMonth === 2;
})->count();

$week3 = $visits->filter(function ($visit) {
    return Carbon::parse($visit->tanggal)->weekOfMonth === 3;
})->count();

$week4 = $visits->filter(function ($visit) {
    return Carbon::parse($visit->tanggal)->weekOfMonth === 4;
})->count();

$thisMonth = $visits->count();

return [
    'week1' => $week1,
    'week2' => $week2,
    'week3' => $week3,
    'week4' => $week4,
    'this_month' => $thisMonth,
];
}
}

```

### 3. JadwalController.php

```

<?php

namespace App\Http\Controllers\Admin;

use Session;
use App\Models\Dokter;
use App\Models\DetailJadwal;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use App\Http\Controllers\Controller;
use Illuminate\Support\Facades\Auth;
use RealRashid\SweetAlert\Facades\Alert;

class JadwalController extends Controller
{
    protected $datadokter;

    public function __construct(DetailJadwal $datadokter)
    {
        $this->datadokter = $datadokter;
    }

    public function index($id)
    {
        $dokter = Dokter::where("dokter_id", $id)->first();
        $dokterjadwal = DetailJadwal::with(['dokter'])->where("dokter_id", $id)->first();
        $datajadwal = DB::table('detail_jadwal')->where('dokter_id', $id)->get();

        $user = Auth::user();
        if ($user->role === 'owner') {
            return view('owner.datadokterjadwal', ['datajadwal' => $datajadwal, 'datadokter' => $dokter]);
        }
    }
}

```

```

        return view('admin.datadokterjadwal', ['datajadwal' => $datajadwal, 'datadokter' =>
$dokter]);
        // $datadokter = [
        //     'datadokter' => $this->datadokter->allData(),
        // ];
    }

    // return view('admin.datadokter', $datadokter);
}

public function add($id)
{
    return view('admin.datadokterjadwal_add', ['id' => $id]);
}

public function insert(Request $request, $id)
{
    // Validasi input form
    $request->validate([
        'hari' => 'required',
        'sesi' => 'required',
    ], [
        'hari.required' => 'Hari harus diisi!',
        'sesi.required' => 'Sesi harus diisi!',
    ]);
}

// Menyiapkan data yang akan disimpan
$data = [
    'hari' => $request->hari,
    'sesi' => $request->sesi,
    'dokter_id' => $id,
];

// Menyimpan data menggunakan model (asumsi ada method addData di model yang
digunakan)
$this->datadokter->addData($data);

// Menampilkan pesan sukses menggunakan SweetAlert (asumsi Alert diimport dengan
benar)
Alert::success('Berhasil!', 'Data jadwal berhasil ditambahkan!');

// Redirect ke halaman yang diinginkan dengan sintaks kurung keriting ganda
return redirect("/datadokter/jadwal/{$id}");
}

public function destroy($jadwal_id)
{
    // Hapus entri terkait di tabel detail_jadwal terlebih dahulu
    DB::table('detail_jadwal')->where('jadwal_id', $jadwal_id)->delete();

    // Menampilkan alert sukses dan mengarahkan kembali ke halaman data dokter
    Alert::success('Berhasil!', 'Data jadwal dokter berhasil dihapus!');
    $id_dokter = DB::table('detail_jadwal')->where('jadwal_id', $jadwal_id)->
value('dokter_id');
    if($id_dokter==NULL){
        return redirect("/datadokter");
    }
    else{
        return redirect("/datadokter/jadwal/{$id_dokter}");
    }
}

public function edit($id)
{
    $datadokter = DB::table('detail_jadwal')->where('jadwal_id', $id)->first();
    return view('admin.datadokterjadwal_edit', ['datadokter' => $datadokter]);
}

public function update(Request $request, $id)
{
    $request->validate([
        'hari' => 'required',
        'sesi' => 'required',
    ], [
        'hari.required' => 'Hari harus diisi!',
        'sesi.required' => 'Sesi harus diisi!',
    ]);
}

$data = [
    'hari' => $request->hari,

```

```

'sesi' => $request->sesi,
];

DB::table('detail_jadwal')->where('jadwal_id', $id)->update($data);
$id_dokter = DB::table('detail_jadwal')->where('jadwal_id', $id)->value('dokter_id');
Alert::success('Berhasil!', 'Data Jadwal berhasil diupdate!');
return redirect("/datadokter/jadwal/{$id_dokter}");
}
}

```

#### 4. PasienController.php

```

<?php

namespace App\Http\Controllers\Admin;

use Session;
use Illuminate\Http\Request;
use App\Models\Pasien;
use Illuminate\Support\Facades\DB;
use RealRashid\SweetAlert\Facades\Alert;
use App\Http\Controllers\Controller;
use Illuminate\Support\Facades\Auth;

class PasienController extends Controller
{
    protected $pasien;

    public function __construct(Pasien $pasien)
    {
        $this->pasien = $pasien;
    }

    public function index()
    {
        $reservasi = DB::table('reservasi_rekam_medik')
            ->join('pasien', 'reservasi_rekam_medik.pasien_id', '=', 'pasien.pasien_id')
            ->join('lokasi', 'reservasi_rekam_medik.lokasi_id', '=', 'lokasi.lokasi_id')
            ->join("dokter", "reservasi_rekam_medik.dokter_id", "=", "dokter.dokter_id")
            ->where('draft', 0)
            ->select('reservasi_rekam_medik.*',
                    'lokasi.*',
                    "pasien.*",
                    'dokter.nama as nama_dokter')->get();

        $user = Auth::user();
        if ($user->role === 'owner') {
            return view('owner.datapasien', compact('reservasi'));
        } elseif($user->role === 'dokter' || $user->role === 'Dokter') {
            return view('dokter.datapasien', compact('reservasi'));
        }
        else{
            return view('admin.datapasien', compact('reservasi'));
        }
    }

    public function add()
    {
        $user = Auth::user();
        if($user->role === 'dokter' || $user->role === 'Dokter') {
            return view('dokter.datapasien_add');
        }
        else{
            return view('admin.datapasien_add');
        }
    }

    public function insert(Request $request)
    {
        $request->validate([
            'nama' => 'required',
            'tempat_lahir' => 'required',
            'tanggal_lahir' => 'required',
            'pekerjaan' => 'required',
            'alamat' => 'required',
            'no_Telp' => 'required',
        ], [

```

```

        'nama.required' => 'Nama harus diisi!',
        'tempat_lahir.required' => 'tempat_lahir harus diisi!',
        'tanggal_lahir.required' => 'tanggal_lahir harus diisi!',
        'pekerjaan.required' => 'pekerjaan harus diisi!',
        'alamat.required' => 'alamat harus diisi!',
        'no_Telp.required' => 'no_Telp harus diisi!',
    ]);

    $data = [
        'nama' => $request->nama,
        'tempat_lahir' => $request->tempat_lahir,
        'tanggal_lahir' => $request->tanggal_lahir,
        'pekerjaan' => $request->pekerjaan,
        'alamat' => $request->alamat,
        'no_Telp' => $request->no_Telp,
    ];

    $this->pasien->addData($data);
    Alert::success('Berhasil!', 'Data pasien berhasil ditambahkan!');
    $user = Auth::user();
    if($user->role === 'dokter' || $user->role === 'Dokter') {
        return redirect('/ddatapasien');
    }
    else{
        return redirect('/adatapasien');
    }
}

public function destroy($pasien_id)
{
    $reservasi = DB::table('reservasi_rekam_medik')
        ->join('pasien', 'reservasi_rekam_medik.pasien_id', '=', 'pasien.pasien_id')
        ->join('lokasi', 'reservasi_rekam_medik.lokasi_id', '=', 'lokasi.lokasi_id')
        ->join("dokter", "reservasi_rekam_medik.dokter_id", "=", "dokter.dokter_id")
        ->where('draft', 0)
        ->select('reservasi_rekam_medik.*',
        'lokasi.*',
        'pasien.*',
        'dokter.nama as nama_dokter')->get();

    // Setelah itu, hapus entri dari tabel pasien
    DB::table('pasien')->where('pasien_id', $pasien_id)->delete();

    // Menampilkan alert sukses dan mengarahkan kembali ke halaman data pasien
    Alert::success('Berhasil!', 'Data pasien berhasil dihapus!');
    $user = Auth::user();
    if($user->role === 'dokter' || $user->role === 'Dokter') {
        return redirect('/ddatapasien');
    }
    else{
        return redirect('/adatapasien');
    }
}

public function edit($id)
{
    $pasien = DB::table('pasien')->where('pasien_id', $id)->first();

    $user = Auth::user();
    if($user->role === 'dokter' || $user->role === 'Dokter') {
        return view('dokter.datapasien_edit', ['pasien' => $pasien]);
    }
    else{
        return view('admin.datapasien_edit', ['pasien' => $pasien]);
    }
}

public function update(Request $request, $id)
{
    $request->validate([
        'nama' => 'required',
        'tempat_lahir' => 'required',
        'tanggal_lahir' => 'required',
        'pekerjaan' => 'required',
        'alamat' => 'required',
        'no_Telp' => 'required',
    ]);
}

```

```

    $data = [
        'nama' => $request->nama,
        'tempat_lahir' => $request->tempat_lahir,
        'tanggal_lahir' => $request->tanggal_lahir,
        'pekerjaan' => $request->pekerjaan,
        'alamat' => $request->alamat,
        'no_Telp' => $request->no_Telp,
    ];

    DB::table('pasien')->where('pasien_id', $id)->update($data);
    Alert::success('Berhasil!', 'Data pasien berhasil diupdate!');
    $user = Auth::user();
    if($user->role === 'dokter' || $user->role === 'Dokter') {
        return redirect('/ddatapasien');
    }
    else{
        return redirect('/adatapasien');
    }
}
}

```

## 5. RekamController.php

```

<?php

namespace App\Http\Controllers\Admin;

use Session;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use App\Models\RekamMedik;
use App\Models\Reservasi;
use App\Models\PerawatanReservasi;
use Illuminate\Support\Facades\DB;
use RealRashid\SweetAlert\Facades\Alert;
use App\Http\Controllers\Controller;
use App\Models\Perawatan;

class RekamController extends Controller
{
    protected $rekam_medik;

    public function __construct(RekamMedik $rekam_medik)
    {
        $this->rekam_medik = $rekam_medik;
    }

    public function index($id)
    {
        // Cari semua pasien terkait dengan dokter_id melalui rekam_medik
        $reservasiIds = DB::table('reservasi_rekam_medik')->where('reservasi_id', $id)->pluck('pasien_id');

        $reservasi_dokter = DB::table("reservasi_rekam_medik")
            ->join("dokter", "reservasi_rekam_medik.dokter_id", "=", "dokter.dokter_id")
            ->where("reservasi_id", $id)
            ->first();

        $reservasi_pasien = DB::table("reservasi_rekam_medik")
            ->join("pasien", "reservasi_rekam_medik.pasien_id", "=", "pasien.pasien_id")
            ->where("reservasi_id", $id)
            ->first();

        $perawatan_reservasi = PerawatanReservasi::with(['reservasi', 'perawatan'])
            ->where('reservasi_id', $id )
            ->get();

        // $rekam_medik = DB::table('rekam_medik')
        // ->whereIn('rekam_medik.reservasi_id', $reservasiIds)
        // ->first();

        // $rekam_medik_perawatan = DB::table('rekam_medik')
        // ->join('perawatan', 'rekam_medik.perawatan_id', '=', 'perawatan.perawatan_id')
        // ->whereIn('rekam_medik.reservasi_id', $reservasiIds)
        // ->first();

        $reservasi_rekam_medik = Reservasi::with('perawatan')->findOrFail($id);
        $user = Auth::user();
        if ($user->role === 'owner') {
            return view('owner.rekammedik', compact('reservasi_dokter', 'reservasi_pasien'
            , 'reservasi_rekam_medik', 'perawatan_reservasi'));
        }
    }
}

```

```

        }
        elseif($user->role === 'dokter' || $user->role === 'Dokter') {
            return view('dokter.rekammedik', compact('reservasi_dokter', 'reservasi_pasien',
            'reservasi_rekam_medik', 'perawatan_reservasi'));
        }
        else{
            return view('admin.rekammedik', compact('reservasi_dokter', 'reservasi_pasien',
            'reservasi_rekam_medik', 'perawatan_reservasi'));
        }
        // $rekam_medik = [
        //     'rekam_medik' => $this->rekam_medik->allData(),
        // ];
        // return view('admin.rekam_medik', $rekam_medik);
    }

    public function add($id)
    {
        $user = Auth::user();
        if($user->role === 'dokter' || $user->role === 'Dokter') {
            return view('dokter.rekam_medik_add', ['id' => $id]);
        }
        else
        {
            return view('admin.rekam_medik_add', ['id' => $id]);
        }
    }

    public function edit($id)
    {
        $perawatan = DB::table('perawatan')->get();

        $reservasi = DB::table('reservasi_rekam_medik')->where('reservasi_id', $id)->first();
        $user = Auth::user();

        if($user->role === 'dokter' || $user->role === 'Dokter') {
            return view('dokter.rekammedik_edit', ['reservasi' => $reservasi],
            compact('perawatan'));
        }
        else
        {
            return view('admin.rekammedik_edit', ['reservasi' => $reservasi],
            compact('perawatan'));
        }
    }

    public function update(Request $request, $id)
    {
        // Validasi data yang dikirimkan dari formulir
        $validatedData = $request->validate([
            'golongan_darah' => 'required',
            'tekanan_darah' => 'required',
            'penyakit_jantung' => 'required',
            'diabetes' => 'required',
            'hepatitis' => 'required',
            'penyakit_lainnya' => 'required',
            'alergi_makanan' => 'required',
            'alergi_obat' => 'required',
            'keluhan' => 'required',
            'perawatan_id' => 'required',
            'gigi' => 'required',
            'perawatan_id' => 'required|array|max:2',
            'perawatan_id.*' => 'integer|exists:perawatan,perawatan_id',
        ]);

        // Perbarui data rekam medik dengan data yang dikirimkan dari formulir

        // Mengambil data harga dan estimasi waktu dari tabel perawatan berdasarkan ID

        $syncData = [];
        foreach ($validatedData['perawatan_id'] as $perawatanId) {
            $perawatan = Perawatan::find($perawatanId);
            $syncData[$perawatanId] = [
                'harga' => $perawatan->harga,
                'estimasi_waktu_perawatan' => $perawatan->estimasi_waktu_perawatan,
            ];
        }
    }
}

```

```

    $reservasi = Reservasi::findOrFail($id);
    $reservasi->update($validatedData);
    $reservasi->perawatan()->sync($syncData);

    Alert::success('Berhasil!', 'Data dokter berhasil diupdate!');
    $user = Auth::user();
    if($user->role === 'dokter' || $user->role === 'Dokter') {
        return redirect()->route("dokter.rekampasien",[$id]);
    }
    else
    {
        return redirect()->route("admin.rekampasien",[$id]);
    }

}
}

```

## 6. ReservasiController.php

```

<?php

namespace App\Http\Controllers\Admin;

use Session;
use App\Models\Pasien;
use Illuminate\Http\Request;
use App\Models\User;
use App\Models\Lokasi;
use App\Models\Perawatan;
use App\Models\Reservasi;
use App\Models\RekamMedik;
use App\Models\Dokter;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Hash;
use RealRashid\SweetAlert\Facades\Alert;
use App\Http\Controllers\Controller;
use App\Models\DetailJadwal;
use Illuminate\Support\Carbon;
use App\Services\WhatsAppService;
use Exception;
use Illuminate\Support\Facades\Log;
use Illuminate\Validation\ValidationException;
use DateTime;
use DateInterval;

class ReservasiController extends Controller
{
    protected $user;
    protected $whatsappService;

    public function __construct(User $user, WhatsAppService $whatsappService)
    {
        $this->user = $user;
        $this->whatsappService = $whatsappService;
    }

    public function index()
    {

        $reservasi_rekam_medik = DB::table('reservasi_rekam_medik')
            ->join('pasien', 'reservasi_rekam_medik.pasien_id', '=', 'pasien.pasien_id')
            ->join('lokasi', 'reservasi_rekam_medik.lokasi_id', '=', 'lokasi.lokasi_id')
            ->join("dokter", "reservasi_rekam_medik.dokter_id", "=", "dokter.dokter_id")
            ->where('draft', 0)
            ->select('reservasi_rekam_medik.*',
            'lokasi.*',
            'pasien.*',
            'dokter.nama as nama_dokter')
            ->orderBy('tanggal', 'desc')
            ->get();

        $reservasi_admin = DB::table("reservasi_rekam_medik")
            ->join("pasien", "reservasi_rekam_medik.pasien_id", "=", "pasien.pasien_id")->get();

        return view('admin.reservasiadmin', compact('reservasi_rekam_medik'));
    }
}

```

```

public function pasien()
{
    $reservasi_rekam_medik = DB::table('reservasi_rekam_medik')
        ->join('pasien', 'reservasi_rekam_medik.pasien_id', '=', 'pasien.pasien_id')
        ->join('lokasi', 'reservasi_rekam_medik.lokasi_id', '=', 'lokasi.lokasi_id')
        ->join("dokter", "reservasi_rekam_medik.dokter_id", "=", "dokter.dokter_id")
        ->where('draft', NULL)
        ->select('reservasi_rekam_medik.*',
            'lokasi.*',
            'pasien.*',
            'dokter.nama as nama_dokter')->get();

    return view('admin.reservasipasien', compact('reservasi_rekam_medik'));
}

public function add()
{
    $lokasi = Lokasi::all();
    $perawatan = Perawatan::all();
    $dokter = Dokter::all();

    $tomorrow = Carbon::tomorrow()->format('Y-m-d');

    // Generate available timeslots
    $timeslots = [];
    for ($hour = 10; $hour < 21; $hour++) {
        for ($minute = 0; $minute < 60; $minute += 30) {
            $start = Carbon::createFromTime($hour, $minute);
            $end = $start->copy()->addMinutes(30);
            $timeslots[] = [
                'start' => $start->format('H:i'),
                'end' => $end->format('H:i')
            ];
        }
    }

    $user = Auth::user();
    if ($user->role === 'Pasien') {
        return view('pasien.reservasipasien_add', compact('lokasi', 'dokter', 'perawatan',
            'tomorrow', 'timeslots'));
    } elseif($user->role === 'Admin' || $user->role === 'admin') {
        return view('admin.reservasiadmin_add', compact('lokasi', 'dokter', 'perawatan',
            'tomorrow', 'timeslots'));
    }
}

public function insert(Request $request)
{
    try {
        $validatedData = $request->validate([
            'nama' => 'required|string|max:255',
            'tempat_lahir' => 'required|string|max:255',
            'tanggal_lahir' => 'required|date',
            'pekerjaan' => 'required|string|max:255',
            'alamat' => 'required|string',
            'no_Telp' => 'required|string|max:15',
            'lokasi_id' => 'required|integer|exists:lokasi,lokasi_id',
            'dokter_id' => 'required|integer',
            'tanggal' => 'required|date',
            'jam_mulai' => 'required',
            'perawatan_id' => 'required|array|max:2',
            'perawatan_id.*' => 'integer|exists:perawatan,perawatan_id',
        ]);
        $tanggal = Carbon::parse($validatedData['tanggal'])->format('Y-m-d');

        // Calculate total estimated time
        $totalEstimasi = 0;
        foreach ($validatedData['perawatan_id'] as $perawatanId) {
            $perawatan = Perawatan::find($perawatanId);
            $totalEstimasi += $perawatan->estimasi_waktu_perawatan;
        }

        $startTime = new DateTime($validatedData['jam_mulai']);
        $endTime = ($clone $startTime)->modify("+{$totalEstimasi} minutes")->format('H:i');

        // Check for time conflicts
    }
}

```

```

$conflictingReservation = Reservasi::where('tanggal', $tanggal)
    ->where(function ($query) use ($validatedData, $endTime) {
        $query->whereBetween('jam_mulai', [$validatedData['jam_mulai'], $endTime])
            ->orWhereBetween('jam_selesai', [$validatedData['jam_mulai'],
$endTime])
                ->orWhereRaw('? BETWEEN jam_mulai AND jam_selesai',
[$validatedData['jam_mulai']]);
            ->orWhereRaw('? BETWEEN jam_mulai AND jam_selesai', [$endTime]);
        })
    ->exists();

// Create new reservation

$pasien = Pasien::create([
    'nama' => $validatedData['nama'],
    'tempat_lahir' => $validatedData['tempat_lahir'],
    'tanggal_lahir' => $validatedData['tanggal_lahir'],
    'pekerjaan' => $validatedData['pekerjaan'],
    'alamat' => $validatedData['alamat'],
    'no_Telp' => $validatedData['no_Telp'],
]);

$reservasi_rekam_medis = Reservasi::create([
    'pasien_id' => $pasien['pasien_id'],
    'lokasi_id' => $validatedData['lokasi_id'],
    'dokter_id' => $validatedData['dokter_id'],
    'tanggal' => $tanggal,
    'jam_mulai' => $validatedData['jam_mulai'],
    'jam_selesai' => $endTime,
    'status_penginput' => 1,
    'draft' => 0
]);

$syncData = [];
foreach ($validatedData['perawatan_id'] as $perawatanId) {
    $perawatan = Perawatan::find($perawatanId);
    $totalEstimasi += $perawatan->estimasi_waktu_perawatan;
    $syncData[$perawatanId] = [
        'harga' => $perawatan->harga,
        'estimasi_waktu_perawatan' => $perawatan->estimasi_waktu_perawatan,
    ];
}

$reservasi_rekam_medis->perawatan()->attach($syncData);

Alert::success('Success', 'Reservasi berhasil ditambahkan!');
return redirect()->route('reservasi.index');
} catch (Exception $e) {
    dd('simpan data gagal : ', $e);
}
}

public function getBookedTimes(Request $request)
{
    $dokter_id = $request->query('dokter_id');
    $tanggal = $request->query('tanggal');
    $tanggalDate = Carbon::parse($tanggal);

    $bookedTimes = Reservasi::where('dokter_id', $dokter_id)
        ->where('tanggal', $tanggalDate)
        ->get(['jam_mulai', 'jam_selesai']);

    return response()->json($bookedTimes);
}

public function getDokterByLokasi(Request $request)
{
    $lokasi_id = $request->query('lokasi_id');
    $dokter = Dokter::where('lokasi_id', $lokasi_id)->get();
    return response()->json($dokter);
}

public function getAvailableDays(Request $request)
{
    $dokter_id = $request->query('dokter_id');
    $jadwals = DetailJadwal::where('dokter_id', $dokter_id)->pluck('hari');
    return response()->json($jadwals);
}

```

```

}

public function getSesiByDokterAndHari(Request $request)
{
    $dokter_id = $request->query('dokter_id');
    $hari = $request->query('hari');
    $jadwals = DetailJadwal::where('dokter_id', $dokter_id)
        ->where('hari', $hari)
        ->pluck('sesi');
    return response()->json($jadwals);
}

public function destroy($id)
{
    // Hapus entri terkait di tabel detail_jadwal terlebih dahulu
    DB::table('reservasi_rekam_medik')->where('reservasi_id', $id)->delete();

    // Menampilkan alert sukses dan mengarahkan kembali ke halaman data user
    Alert::success('Berhasil!', 'Data reservasi_rekam_medik berhasil dihapus!');
    return redirect('/reservasi/admin');
}

public function terima($id)
{
    $reservasi_rekam_medik = Reservasi::where('reservasi_id', $id)->firstOrFail();

    // Prepare data for reservasi_rekam_medik
    $reservasiData = [
        'draft' => 0,
    ];
    // Update the reservasi_rekam_medik
    $reservasi_rekam_medik->update($reservasiData);
    $reservasi = Reservasi::with(['pasien'])
    ->where('reservasi_id', $id)
    ->firstOrFail();

    $this->whatsappService->sendWhatsAppMessage($reservasi->pasien->no_Telp, 'Reservasi
Anda telah diterima, silahkan cek status reservasi anda pada akun website Xenon Dental House
anda');

    // Redirect or return a response
    Alert::success('Berhasil!', 'Data reservasi_rekam_medik pasien diterima !');
    return redirect()->route('reservasi.pasien');
}

public function tolak($id)
{
    $reservasi_rekam_medik = Reservasi::where('reservasi_id', $id)->firstOrFail();

    // Prepare data for reservasi_rekam_medik
    $reservasiData = [
        'draft' => 1,
    ];
    // Update the reservasi_rekam_medik
    $reservasi_rekam_medik->update($reservasiData);

    $reservasi = Reservasi::with(['pasien'])
    ->where('reservasi_id', $id)
    ->firstOrFail();

    $this->whatsappService->sendWhatsAppMessage($reservasi->pasien->no_Telp, 'Reservasi
Anda telah ditolak, silahkan cek status reservasi anda pada akun website Xenon Dental House
anda');

    // Redirect or return a response
    Alert::success('Berhasil!', 'Data reservasi_rekam_medik pasien ditolak !');
    return redirect()->route('reservasi.pasien');
}

public function edit($id)
{
    $lokasi = Lokasi::all();
    $perawatan = Perawatan::all();
    $dokter = Dokter::all();
    $reservasi = Reservasi::with(['pasien', 'dokter', 'lokasi']);
}

```

```

        ->where('reservasi_id', $id)
        ->first();

        $tomorrow = Carbon::tomorrow()->format('Y-m-d');

        // Generate available timeslots
        $timeslots = [];
        for ($hour = 10; $hour < 21; $hour++) {
            for ($minute = 0; $minute < 60; $minute += 30) {
                $start = Carbon::createFromTime($hour, $minute);
                $end = $start->copy()->addMinutes(30);
                $timeslots[] = [
                    'start' => $start->format('H:i'),
                    'end' => $end->format('H:i')
                ];
            }
        }

        return view('admin.reservasiadmin_edit', compact('lokasi', 'dokter',
        'perawatan','reservasi', 'tomorrow'));
    }

    public function update(Request $request, $reservasi_id)
    {
        try {
            $validatedData = $request->validate([
                'nama' => 'required|string|max:255',
                'tempat_lahir' => 'required|string|max:255',
                'tanggal_lahir' => 'required|date',
                'pekerjaan' => 'required|string|max:255',
                'alamat' => 'required|string',
                'no_Telp' => 'required|string|max:15',
                'lokasi_id' => 'required|integer|exists:lokasi,lokasi_id',
                'dokter_id' => 'required|integer',
                'tanggal' => 'required|date',
                'jam_mulai' => 'required',
                'perawatan_id' => 'required|array|max:2',
                'perawatan_id.*' => 'integer|exists:perawatan,perawatan_id',
            ]);
        }

        $tanggal = Carbon::parse($validatedData['tanggal'])->format('Y-m-d');

        // Calculate total estimated time
        $totalEstimasi = 0;
        foreach ($validatedData['perawatan_id'] as $perawatanId) {
            $perawatan = Perawatan::find($perawatanId);
            $totalEstimasi += $perawatan->estimasi_waktu_perawatan;
        }

        $startTime = new DateTime($validatedData['jam_mulai']);
        $endTime = (clone $startTime)->modify("+{$totalEstimasi} minutes")->format('H:i');

        // Check for time conflicts
        $conflictingReservation = Reservasi::where('tanggal', $tanggal)
            ->where('reservasi_id', '!=', $reservasi_id)
            ->where(function ($query) use ($validatedData, $endTime) {
                $query->whereBetween('jam_mulai', [$validatedData['jam_mulai'], $endTime])
                    ->orWhereBetween('jam_selesai', [$validatedData['jam_mulai'],
                    $endTime])
                    ->orWhereRaw('? BETWEEN jam_mulai AND jam_selesai',
                    [$validatedData['jam_mulai']])
                    ->orWhereRaw('? BETWEEN jam_mulai AND jam_selesai', [$endTime]);
            })
            ->exists();

        // if ($conflictingReservation) {
        //     Alert::error('Error', 'Waktu yang dipilih bentrok dengan reservasi lain!');
        //     return redirect()->back()->withInput();
        // }

        // Update existing reservation
        $reservasi = Reservasi::findOrFail($reservasi_id);
        $pasien = Pasien::findOrFail($reservasi->pasien_id);

        $pasien->update([
            'nama' => $validatedData['nama'],
            'tempat_lahir' => $validatedData['tempat_lahir'],
            'tanggal_lahir' => $validatedData['tanggal_lahir'],
            'pekerjaan' => $validatedData['pekerjaan'],
        ]);
    }
}

```

```

        'alamat' => $validatedData['alamat'],
        'no_Telp' => $validatedData['no_Telp'],
    ]);

    $reservasi->update([
        'lokasi_id' => $validatedData['lokasi_id'],
        'dokter_id' => $validatedData['dokter_id'],
        'tanggal' => $tanggal,
        'jam_mulai' => $validatedData['jam_mulai'],
        'jam_selesai' => $endTime,
        'status_penginput' => 1,
        'draft' => 0
    ]);

    $syncData = [];
    foreach ($validatedData['perawatan_id'] as $perawatanId) {
        $perawatan = Perawatan::find($perawatanId);
        $syncData[$perawatanId] = [
            'harga' => $perawatan->harga,
            'estimasi_waktu_perawatan' => $perawatan->estimasi_waktu_perawatan,
        ];
    }

    $reservasi->perawatan()->sync($syncData);

    Alert::success('Success', 'Reservasi berhasil diperbarui!');
    return redirect()->route('reservasi.index');
} catch (Exception $e) {
    dd('update data gagal : ', $e);
}
}

}

```

## 7. UserController.php

```

<?php

namespace App\Http\Controllers\Admin;

use Session;
use App\Models\User;
use App\Models\Dokter;
use App\Models\Lokasi;
use App\Models\DataDokter;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use App\Http\Controllers\Controller;
use Illuminate\Support\Facades\Hash;
use RealRashid\SweetAlert\Facades\Alert;

class UserController extends Controller
{
    protected $user;

    public function __construct(User $user)
    {
        $this->user = $user;
    }

    public function index()
    {
        $user = [
            'user' => $this->user->allData(),
        ];

        return view('admin.datauser', $user);
    }

    public function add()
    {
        return view('admin.datauser_add');
    }

    public function insert(Request $request)
    {
        $request->validate([
            'name' => 'required',
            'email' => 'required',
            'password' => 'required',
            'role' => 'required',
        ]);
    }
}

```

```

], [
    'name.required' => 'Nama harus diisi!',
    'email.required' => 'email harus diisi!',
    'password.required' => 'role harus diisi!',
    'role.required' => 'role harus diisi!',
];

$data = [
    'name' => $request->name,
    'email' => $request->email,
    'password' => Hash::make($request->password),
    'role' => $request->role,
];

$this->user->addData($data);
Alert::success('Berhasil!', 'Data user berhasil ditambahkan!');
return redirect('/datauser');
}

public function destroy($id)
{
    // Hapus entri terkait di tabel detail_jadwal terlebih dahulu
    DB::table('users')->where('id', $id)->delete();

    // Menampilkan alert sukses dan mengarahkan kembali ke halaman data user
    Alert::success('Berhasil!', 'Data user berhasil dihapus!');
    return redirect('/datauser');
}

public function add_dokter()
{
    $lokasi = Lokasi::all();
    return view('admin.datauserdokter_add', compact('lokasi'));
}

public function insert_dokter(Request $request)
{
    $request->validate([
        'nama' => 'required',
        'alamat' => 'required',
        'nomor_hp' => 'required',
        'lokasi_id' => 'required',
        'email' => 'required',
        'password' => 'required',
    ], [
        'nama.required' => 'Nama harus diisi!',
        'alamat.required' => 'alamat harus diisi!',
        'nomor_hp.required' => 'nomor_hp harus diisi!',
        'lokasi_id.required' => 'lokasi praktek harus diisi',
        'email.required' => 'email harus diisi!',
        'password.required' => 'role harus diisi!',
    ]);

    $data = [
        'name' => $request->nama,
        'email' => $request->email,
        'password' => Hash::make($request->password),
        'role' => 'Dokter',
    ];

    $id_user = $this->user->addData($data);

    Dokter::create([
        'nama' => $request->nama,
        'alamat' => $request->alamat,
        'nomor_hp' => $request->nomor_hp,
        'lokasi_id' => $request->lokasi_id,
        'user_id' => $id_user
    ]);

    Alert::success('Berhasil!', 'Data akun dokter berhasil ditambahkan!');
    return redirect('/datauser');
}

```

```

public function edit($id)
{
    $user = DB::table('users')->where('id', $id)->first();
    return view('admin.datauser_edit', ['user' => $user]);
}

public function update(Request $request, $id)
{
    $request->validate([
        'name' => 'required',
        'email' => 'required',
        'password' => 'required',
        'role' => 'required',
    ]);

    $data = [
        'name' => $request->name,
        'email' => $request->email,
        'password' => $request->password,
        'role' => $request->role,
    ];

    DB::table('users')->where('id', $id)->update($data);
    Alert::success('Berhasil!', 'Data user berhasil diupdate!');
    return redirect('/datauser');
}
}

```

### Kode Controller Auth

#### 1. AuthenticatedSessionController.php

```

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Http\Requests\Auth\LoginRequest;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\View\View;

class AuthenticatedSessionController extends Controller
{
    /**
     * Display the login view.
     */
    public function create(): View
    {
        return view('auth.login');
    }

    /**
     * Handle an incoming authentication request.
     */
    public function store(LoginRequest $request)
    {
        $request->authenticate();

        $request->session()->regenerate();

        $user = Auth::user();

        if (!$user) {
            return response()->json(['success' => false, 'message' => 'Authentication failed.'], 401);
        }

        $role = $user->role;
        $redirectUrl = '/admin'; // Default redirect

        if ($role === 'owner' || $role === 'Owner') {
            return redirect('/owner');
        } elseif ($role === 'dokter' || $role === 'Dokter') {
            return redirect('/dokter');
        } elseif ($role === 'pasien' || $role === 'Pasien') {
            return redirect('/home');
        }
    }
}

```

```

        return redirect("/admin");
    }

    /**
     * Destroy an authenticated session.
     */
    public function destroy(Request $request): RedirectResponse
    {
        Auth::guard('web')->logout();

        $request->session()->invalidate();

        $request->session()->regenerateToken();

        return redirect('/');
    }
}

```

## 2. EmailVerificationPromptController.php

```

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Http\Requests\Auth\LoginRequest;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\View\View;

class AuthenticatedSessionController extends Controller
{
    /**
     * Display the login view.
     */
    public function create(): View
    {
        return view('auth.login');
    }

    /**
     * Handle an incoming authentication request.
     */
    public function store(LoginRequest $request)
    {
        $request->authenticate();

        $request->session()->regenerate();

        $user = Auth::user();

        if (!$user) {
            return response()->json(['success' => false, 'message' => 'Authentication failed.'], 401);
        }

        $role = $user->role;
        $redirectTo = '/admin'; // Default redirect

        if ($role === 'owner' || $role === 'Owner') {
            return redirect('/owner');
        } elseif ($role === 'dokter' || $role === 'Dokter') {
            return redirect('/dokter');
        } elseif ($role === 'pasien' || $role === 'Pasien') {
            return redirect('/home');
        }

        return redirect("/admin");
    }

    /**
     * Destroy an authenticated session.
     */
    public function destroy(Request $request): RedirectResponse
    {
        Auth::guard('web')->logout();
    }
}

```

```

        $request->session()->invalidate();

        $request->session()->regenerateToken();

        return redirect('/');
    }
}

```

### 3. NewPasswordController.php

```

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use Illuminate\Auth\Events\PasswordReset;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
use Illuminate\Support\Facades\Password;
use Illuminate\Support\Str;
use Illuminate\Validation\Rules;
use Illuminate\View\View;

class NewPasswordController extends Controller
{
    /**
     * Display the password reset view.
     */
    public function create(Request $request): View
    {
        return view('auth.reset-password', ['request' => $request]);
    }

    /**
     * Handle an incoming new password request.
     *
     * @throws \Illuminate\Validation\ValidationException
     */
    public function store(Request $request): RedirectResponse
    {
        $request->validate([
            'token' => ['required'],
            'email' => ['required', 'email'],
            'password' => ['required', 'confirmed', Rules\Password::defaults()],
        ]);

        // Here we will attempt to reset the user's password. If it is successful we
        // will update the password on an actual user model and persist it to the
        // database. Otherwise we will parse the error and return the response.
        $status = Password::reset(
            $request->only('email', 'password', 'password_confirmation', 'token'),
            function ($user) use ($request) {
                $user->forceFill([
                    'password' => Hash::make($request->password),
                    'remember_token' => Str::random(60),
                ])->save();

                event(new PasswordReset($user));
            }
        );

        // If the password was successfully reset, we will redirect the user back to
        // the application's home authenticated view. If there is an error we can
        // redirect them back to where they came from with their error message.
        return $status == Password::PASSWORD_RESET
            ? redirect()->route('login')->with('status', __($status))
            : back()->withInput($request->only('email'))
                ->withErrors(['email' => __($status)]);
    }
}

```

### 4. PasswordController.php

```

<?php

namespace App\Http\Controllers\Auth;

```

```

use App\Http\Controllers\Controller;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Hash;
use Illuminate\Validation\Rules\Password;

class PasswordController extends Controller
{
    /**
     * Update the user's password.
     */
    public function update(Request $request): RedirectResponse
    {
        $validated = $request->validateWithBag('updatePassword', [
            'current_password' => ['required', 'current_password'],
            'password' => ['required', Password::defaults(), 'confirmed'],
        ]);

        $request->user()->update([
            'password' => Hash::make($validated['password']),
        ]);

        return back()->with('status', 'password-updated');
    }
}

```

## 5. PasswordResetLinkController.php

```

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Password;
use Illuminate\View\View;

class PasswordResetLinkController extends Controller
{
    /**
     * Display the password reset link request view.
     */
    public function create(): View
    {
        return view('auth.forgot-password');
    }

    /**
     * Handle an incoming password reset link request.
     *
     * @throws \Illuminate\Validation\ValidationException
     */
    public function store(Request $request): RedirectResponse
    {
        $request->validate([
            'email' => ['required', 'email'],
        ]);

        // We will send the password reset link to this user. Once we have attempted
        // to send the link, we will examine the response then see the message we
        // need to show to the user. Finally, we'll send out a proper response.
        $status = Password::sendResetLink(
            $request->only('email')
        );

        return $status == Password::RESET_LINK_SENT
            ? back()->with('status', __($status))
            : back()->withInput($request->only('email'))
                ->withErrors(['email' => __($status)]);
    }
}

```

## 6. RegisteredUserController.php

```
<?php
```

```

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Models\User;
use Illuminate\Auth\Events\Registered;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Hash;
use Illuminate\Validation\Rules;
use Illuminate\View\View;

class RegisteredUserController extends Controller
{
    /**
     * Display the registration view.
     */
    public function create(): View
    {
        return view('auth.register');
    }

    /**
     * Handle an incoming registration request.
     *
     * @throws \Illuminate\Validation\ValidationException
     */
    public function store(Request $request): RedirectResponse
    {
        $request->validate([
            'name' => ['required', 'string', 'max:255'],
            'email' => ['required', 'string', 'lowercase', 'email', 'max:255',
            'unique:'.User::class],
            'password' => ['required', 'confirmed', Rules\Password::defaults()],
        ]);

        $user = User::create([
            'name' => $request->name,
            'email' => $request->email,
            'password' => Hash::make($request->password),
            'role' => 'Pasien',
        ]);

        event(new Registered($user));

        Auth::login($user);

        $user = Auth::user();
        if ($user->role === 'owner' || $user->role === 'Owner') {
            return redirect('/owner');
        }
        elseif ($user->role === 'dokter' || $user->role === 'Dokter') {
            return redirect('/dokter');
        }
        elseif ($user->role === 'Pasien' || $user->role === 'pasien') {
            return redirect('/home');
        }

        return redirect("/admin");
    }
}

```

## 7. VerifyEmailController.php

```

<?php

namespace App\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use Illuminate\Auth\Events\Verified;
use Illuminate\Foundation\Auth\EmailVerificationRequest;
use Illuminate\Http\RedirectResponse;

class VerifyEmailController extends Controller
{
    /**
     * Mark the authenticated user's email address as verified.
     */
    public function __invoke(EmailVerificationRequest $request): RedirectResponse
    {

```

```

if ($request->user()->hasVerifiedEmail()) {
    return redirect()->intended(route('admin', absolute: false).'?verified=1');
}

if ($request->user()->markEmailAsVerified()) {
    event(new Verified($request->user()));
}

return redirect()->intended(route('admin', absolute: false).'?verified=1');
}

```

### Kode Controller Dokter

#### 1. DokterHomeController.php

```

<?php

namespace App\Http\Controllers\Dokter;

use App\Http\Controllers\Controller;
use App\Models\DetailJadwal;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use Carbon\Carbon;
use Illuminate\Support\Facades\Auth;
use App\Models\Izin;

class DokterHomeController extends Controller
{
    public function index()
    {
        $user = Auth::user();
        $userid = $user->id;

        // $jadwal = DetailJadwal::with('dokter')
        // ->where('user_id', $userid)
        // ->get();

        $jadwalid = DB::table('detail_jadwal')
            ->join('dokter', 'detail_jadwal.dokter_id', '=', 'dokter.dokter_id')
            ->where('dokter.user_id', $userid)->get();

        // Determine the user role and return the appropriate view

        return view('dokter.home', compact("jadwalid"));

    }
}

```

#### 2. IzinController.php

```

<?php

namespace App\Http\Controllers\Dokter;

use Session;
use Exception;
use App\Models\Izin;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use App\Http\Controllers\Controller;
use Illuminate\Support\Facades\Auth;
use RealRashid\SweetAlert\Facades\Alert;

class IzinController extends Controller
{
    protected $izin;

    public function __construct(izin $izin)
    {
        $this->izin = $izin;
    }
}

```

```

public function index()
{
    $izin = [
        'izin' => $this->izin->allData(),
    ];

    $izin_user = IZIN::with(['users'])->get();
    $izin_user_owner = IZIN::with(['users'])
    ->whereNull('status')
    ->get();

    $user = Auth::user();
    if($user->role === 'Dokter'){
        return view('dokter.dataabsen', compact('izin','izin_user'));
    }
    else{
        return view('owner.dataizin', compact('izin','izin_user_owner'));
    }
}

public function add()
{
    return view('dokter.dataabsen_add');
}

public function insert(Request $request)
{
    try{

        $request->validate([
            'tanggal_awal' => 'required',
            'tanggal_akhir' => 'required',
            'alasan' => 'required',
        ],[
            'tanggal_awal' => 'Tanggal awal harus diisi',
            'tanggal_akhir' => 'tanggal akhir harus diisi',
            'alasan' => 'alasan harus diisi',
        ]);

        $user = Auth::user();
        $data = [
            'tanggal_awal' => $request->tanggal_awal,
            'tanggal_akhir' => $request->tanggal_akhir,
            'alasan' => $request->alasan,
            'user_id' => $user->id,
        ];

        $this->izin->addData($data);
        Alert::success('Berhasil!', 'Data Izin berhasil diajukan!');
        return redirect('/dataabsen');
    } catch (Exception $e) {
        dd('simpan data gagal : ', $e);
    }
}

public function destroy($dokter_id)
{
    // Hapus entri terkait di tabel detail_jadwal terlebih dahulu
    // DB::table('detail_jadwal')->where('dokter_id', $dokter_id)->delete();

    // Hapus entri terkait di tabel izin
    // DB::table('izin')->where('dokter_id', $dokter_id)->delete();

    // Cari semua reservasi terkait dengan dokter_id melalui rekam_medik
    // $reservasiIds = DB::table('rekam_medik')->where('dokter_id', $dokter_id)->pluck('reservasi_id');

    // Hapus entri terkait di tabel rekam_medik
    // DB::table('rekam_medik')->where('dokter_id', $dokter_id)->delete();

    // Hapus entri terkait di tabel reservasi
    // DB::table('reservasi')->whereIn('reservasi_id', $reservasiIds)->delete();

    // Setelah itu, hapus entri dari tabel dokter
    DB::table('izin_dokter')->where('izin_id', $dokter_id)->delete();
}

```

```

// Menampilkan alert sukses dan mengarahkan kembali ke halaman data dokter
Alert::success('Berhasil!', 'Data izin berhasil dihapus!');
return redirect('/dataabsen');
}

public function edit($id)
{
    $izin = DB::table('izin_dokter')->where('izin_id', $id)->first();
    return view('dokter.dataabsen_edit', ['izin' => $izin]);
}

public function detail($id)
{
    $datajadwal = DB::table('detail_jadwal')->where('dokter_id', $id)->get();
    return view('admin.izinjadwal', ['datajadwal' => $datajadwal]);
}

public function tolak($id)
{
    $izin = Izin::where('izin_id', $id)->firstOrFail();

    // Prepare data for reservasi
    $perizinan = [
        'status' => 0,
    ];
    // Update the reservasi
    $izin->update($perizinan);

    // Redirect or return a response
    Alert::success('Berhasil!', 'Data perizinan dokter ditolak !');
    return redirect()->route('owner.izin.index');

}

public function terima($id)
{
    $izin = Izin::where('izin_id', $id)->firstOrFail();

    // Prepare data for reservasi
    $perizinan = [
        'status' => 1,
    ];
    // Update the reservasi
    $izin->update($perizinan);

    // Redirect or return a response
    Alert::success('Berhasil!', 'Data perizinan dokter diterima !');
    return redirect()->route('owner.izin.index');

}

public function update(Request $request, $id)
{
    $request->validate([
        'tanggal_awal' => 'required',
        'tanggal_akhir' => 'required',
        'alasan' => 'required',
    ]);

    $data = [
        'tanggal_awal' => $request->tanggal_awal,
        'tanggal_akhir' => $request->tanggal_akhir,
        'alasan' => $request->alasan,
    ];

    DB::table('izin_dokter')->where('izin_id', $id)->update($data);
    Alert::success('Berhasil!', 'Data pengajuan izin berhasil diupdate!');
    return redirect('/dataabsen');
}
}

```

### Kode Controller Owner

#### 1. PemasukanController.php

```

<?php

namespace App\Http\Controllers\Owner;

use Session;

```

```

use Illuminate\Http\Request;
use App\Models\Reservasi;
use Illuminate\Support\Facades\DB;
use RealRashid\SweetAlert\Facades\Alert;
use App\Http\Controllers\Controller;
use App\Models\Perawatan;
use App\Models\PerawatanReservasi;
use Illuminate\Support\Facades\Auth;

class PemasukanController extends Controller
{
    protected $reservasi;

    public function __construct(reservasi $pemasukan)
    {
        $this->reservasi = $pemasukan;
    }

    public function index()
    {
        $pemasukan = Reservasi::with(['pasien', 'dokter', 'perawatan_reservasi.perawatan'])
            ->where('draft', 0)
            ->get();

        $perawatan_reservasi = PerawatanReservasi::with(['reservasi'])
            ->get();

        $perawatan = Perawatan::get();

        return view('owner.datapemasukan', ["pemasukan"=> $pemasukan, "perawatan_reservasi"=> $perawatan_reservasi, "perawatan"=> $perawatan]);
    }
}

```

## 2. PengeluaranController.php

```

<?php

namespace App\Http\Controllers\Owner;

use Session;
use Illuminate\Http\Request;
use App\Models\Pengeluaran;
use Illuminate\Support\Facades\DB;
use RealRashid\SweetAlert\Facades\Alert;
use App\Http\Controllers\Controller;
use Illuminate\Support\Facades\Auth;

class PengeluaranController extends Controller
{
    protected $pengeluaran;

    public function __construct(pengeluaran $pengeluaran)
    {
        $this->pengeluaran = $pengeluaran;
    }

    public function index()
    {
        $pengeluaran = [
            'pengeluaran' => $this->pengeluaran->allData(),
        ];
        return view('owner.datapengeluaran', $pengeluaran);
    }

    public function add()
    {
        return view('owner.datapengeluaran_add');
    }

    public function insert(Request $request)
    {
        $request->validate([
            'nama_pengeluaran' => 'required',
            'deskripsi_pengeluaran' => 'required',
            'kategori_pengeluaran' => 'required',
            'jumlah_pengeluaran' => 'required',
            'tanggal' => 'required',
        ], [

```

```

        'nama_pengeluaran.required' => 'nama pengeluaran harus diisi!',
        'deskripsi_pengeluaran.required' => 'estimasi waktu pengeluaran harus diisi!',
        'kategori_pengeluaran' => 'jenis pengeluaran harus diisi!',
        'jumlah_pengeluaran' => 'jumlah pengeluaran harus diisi!',
        'tanggal' => 'tanggal harus diisi!',
    ]);

    $user = Auth::user();

    $data = [
        'nama_pengeluaran' => $request->nama_pengeluaran,
        'deskripsi_pengeluaran' => $request->deskripsi_pengeluaran,
        'kategori_pengeluaran' => $request->kategori_pengeluaran,
        'jumlah_pengeluaran' => $request->jumlah_pengeluaran,
        'tanggal' => $request->tanggal,
        'user_id' => $user->id,
    ];

    $this->pengeluaran->addData($data);
    Alert::success('Berhasil!', 'Data pengeluaran berhasil ditambahkan!');
    return redirect('/datapengeluaran');
}

public function destroy($pengeluaran_id)
{
    // Setelah itu, hapus entri dari tabel pengeluaran
    DB::table('pengeluaran')->where('pengeluaran_id', $pengeluaran_id)->delete();

    // Menampilkan alert sukses dan mengarahkan kembali ke halaman data pengeluaran
    Alert::success('Berhasil!', 'Data pengeluaran berhasil dihapus!');
    return redirect('/datapengeluaran');
}

public function edit($id)
{
    $pengeluaran = DB::table('pengeluaran')->where('pengeluaran_id', $id)->first();
    return view('owner.datapengeluaran_edit', ['pengeluaran' => $pengeluaran]);
}

public function update(Request $request, $id)
{
    $request->validate([
        'nama_pengeluaran' => 'required',
        'deskripsi_pengeluaran' => 'required',
        'kategori_pengeluaran' => 'required',
        'jumlah_pengeluaran' => 'required',
        'tanggal' => 'required',
    ]);

    $data = [
        'nama_pengeluaran' => $request->nama_pengeluaran,
        'deskripsi_pengeluaran' => $request->deskripsi_pengeluaran,
        'kategori_pengeluaran' => $request->kategori_pengeluaran,
        'jumlah_pengeluaran' => $request->jumlah_pengeluaran,
        'tanggal' => $request->tanggal,
    ];

    DB::table('pengeluaran')->where('pengeluaran_id', $id)->update($data);
    Alert::success('Berhasil!', 'Data pengeluaran berhasil diupdate!');
    return redirect('/datapengeluaran');
}
}

```

### 3. PerawatanController.php

```

<?php

namespace App\Http\Controllers\Owner;

use Session;
use Illuminate\Http\Request;
use App\Models\Perawatan;
use Illuminate\Support\Facades\DB;
use RealRashid\SweetAlert\Facades\Alert;
use App\Http\Controllers\Controller;
use Illuminate\Support\Facades\Auth;

class PerawatanController extends Controller
{
    protected $perawatan;

```

```

public function __construct(Perawatan $perawatan)
{
    $this->perawatan = $perawatan;
}

public function index()
{
    $perawatan = [
        'perawatan' => $this->perawatan->allData(),
    ];
    return view('owner.datalayanan', $perawatan);
}

public function add()
{
    return view('owner.datalayanan_add');
}

public function insert(Request $request)
{
    $request->validate([
        'jenis_Perawatan' => 'required',
        'estimasi_waktu_perawatan' => 'required',
        'harga' => 'required',
    ], [
        'jenis_Perawatan.required' => 'jenis perawatan harus diisi!',
        'estimasi_waktu_perawatan.required' => 'estimasi waktu perawatan harus diisi!',
        'harga.required' => 'harga harus diisi!',
    ]);
    $data = [
        'jenis_Perawatan' => $request->jenis_Perawatan,
        'estimasi_waktu_perawatan' => $request->estimasi_waktu_perawatan,
        'harga' => $request->harga,
    ];
    $this->perawatan->addData($data);
    Alert::success('Berhasil!', 'Data perawatan berhasil ditambahkan!');
    return redirect('/datalayanan');
}

public function destroy($layanan_id)
{
    // Setelah itu, hapus entri dari tabel perawatan
    DB::table('perawatan')->where('perawatan_id', $layanan_id)->delete();

    // Menampilkan alert sukses dan mengarahkan kembali ke halaman data perawatan
    Alert::success('Berhasil!', 'Data perawatan berhasil dihapus!');
    return redirect('/datalayanan');
}

public function edit($id)
{
    $perawatan = DB::table('perawatan')->where('perawatan_id', $id)->first();
    return view('owner.datalayanan_edit', ['perawatan' => $perawatan]);
}

public function update(Request $request, $id)
{
    $request->validate([
        'jenis_Perawatan' => 'required',
        'estimasi_waktu_perawatan' => 'required',
        'harga' => 'required',
    ]);

    $data = [
        'jenis_Perawatan' => $request->jenis_Perawatan,
        'estimasi_waktu_perawatan' => $request->estimasi_waktu_perawatan,
        'harga' => $request->harga,
    ];
    DB::table('perawatan')->where('perawatan_id', $id)->update($data);
    Alert::success('Berhasil!', 'Data perawatan berhasil diupdate!');
    return redirect('/datalayanan');
}

```

### Kode Controller Pasien

## 1. ReservasiPasienController.php

```
<?php

namespace App\Http\Controllers\Pasien;

use Session;
use App\Models\Pasien;
use Illuminate\Http\Request;
use App\Models\User;
use App\Models\Lokasi;
use App\Models\Perawatan;
use App\Models\Reservasi;
use App\Models\RekamMedik;
use App\Models\Dokter;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Hash;
use RealRashid\SweetAlert\Facades\Alert;
use App\Http\Controllers\Controller;
use App\Models\DetailJadwal;
use Illuminate\Support\Carbon;
use Exception;
use Illuminate\Support\Facades\Log;
use Illuminate\Validation\ValidationException;
use DateTime;
use DateInterval;

class ReservasiPasienController extends Controller
{
    protected $user;

    public function __construct(User $user)
    {
        $this->user = $user;
    }

    public function index()
    {

        $userid = Auth::user();
        $reservasi_rekam_medik = DB::table('reservasi_rekam_medik')
            ->join('pasien', 'reservasi_rekam_medik.pasien_id', '=', 'pasien.pasien_id')
            ->join('lokasi', 'reservasi_rekam_medik.lokasi_id', '=', 'lokasi.lokasi_id')
            ->join("dokter", "reservasi_rekam_medik.dokter_id", "=", "dokter.dokter_id")
            ->where('pasien.user_id', $userid->id)
            ->where('status_penginput', 0)
            ->select('reservasi_rekam_medik.*',
                    'lokasi.*',
                    "pasien.*",
                    'dokter.nama as nama_dokter')->get();

        $reservasi_admin = DB::table("reservasi_rekam_medik")
            ->join("pasien", "reservasi_rekam_medik.pasien_id", "=", "pasien.pasien_id")->get();

        return view('pasien.reservasipasien', compact('reservasi_rekam_medik'));
    }

    public function pasien()
    {

        $reservasi_rekam_medik = DB::table('reservasi_rekam_medik')
            ->join('pasien', 'reservasi_rekam_medik.pasien_id', '=', 'pasien.pasien_id')
            ->join('lokasi', 'reservasi_rekam_medik.lokasi_id', '=', 'lokasi.lokasi_id')
            ->join("dokter", "reservasi_rekam_medik.dokter_id", "=", "dokter.dokter_id")
            ->where('draft', 1)
            ->select('reservasi_rekam_medik.*',
                    'lokasi.*',
                    "pasien.*",
                    'dokter.nama as nama_dokter')->get();

        return view('admin.reservasipasien', compact('reservasi_rekam_medik'));
    }

    public function add()
    {
        $lokasi = Lokasi::all();
        $perawatan = Perawatan::all();
    }
}
```

```

$dokter = Dokter::all();

$tomorrow = Carbon::tomorrow()->format('Y-m-d');

// Generate available timeslots
$timeslots = [];
for ($hour = 10; $hour < 21; $hour++) {
    for ($minute = 0; $minute < 60; $minute += 30) {
        $start = Carbon::createFromTime($hour, $minute);
        $end = $start->copy()->addMinutes(30);
        $timeslots[] = [
            'start' => $start->format('H:i'),
            'end' => $end->format('H:i')
        ];
    }
}

return view('pasien.reservasipasien_add', compact('lokasi', 'dokter', 'perawatan',
'tomorrow', 'timeslots'));
}

public function insert(Request $request)
{
    try {
        $validatedData = $request->validate([
            'nama' => 'required|string|max:255',
            'tempat_lahir' => 'required|string|max:255',
            'tanggal_lahir' => 'required|date',
            'pekerjaan' => 'required|string|max:255',
            'alamat' => 'required|string',
            'no_Telp' => 'required|string|max:15',
            'lokasi_id' => 'required|integer|exists:lokasi,lokasi_id',
            'dokter_id' => 'required|integer',
            'tanggal' => 'required|date|after:today',
            'jam_mulai' => 'required',
            'perawatan_id' => 'required|array|max:2',
            'perawatan_id.*' => 'integer|exists:perawatan,perawatan_id',
        ]);

        $tanggal = Carbon::parse($validatedData['tanggal'])->format('Y-m-d');

        // Calculate total estimated time
        $totalEstimasi = 0;
        foreach ($validatedData['perawatan_id'] as $perawatanId) {
            $perawatan = Perawatan::find($perawatanId);
            $totalEstimasi += $perawatan->estimasi_waktu_perawatan;
        }

        $startTime = new DateTime($validatedData['jam_mulai']);
        $endTime = (clone $startTime)->modify("+{$totalEstimasi} minutes")->format('H:i');

        // Check for time conflicts
        $conflictingReservation = Reservasi::where('tanggal', $tanggal)
            ->where(function ($query) use ($validatedData, $endTime) {
                $query->whereBetween('jam_mulai', [$validatedData['jam_mulai'], $endTime])
                    ->orWhereBetween('jam_selesai', [$validatedData['jam_mulai'],
                    $endTime])
                    ->orWhereRaw('? BETWEEN jam_mulai AND jam_selesai',
                    [$validatedData['jam_mulai']]);
                ->orWhereRaw('? BETWEEN jam_mulai AND jam_selesai', [$endTime]);
            })
            ->exists();

        // Create new reservation
        $user_id = Auth::id();

        $pasien = Pasien::create([
            'nama' => $validatedData['nama'],
            'tempat_lahir' => $validatedData['tempat_lahir'],
            'tanggal_lahir' => $validatedData['tanggal_lahir'],
            'pekerjaan' => $validatedData['pekerjaan'],
            'alamat' => $validatedData['alamat'],
            'no_Telp' => $validatedData['no_Telp'],
            'user_id' => $user_id,
        ]);

        $reservasi_rekam_medik = Reservasi::create([
            'pasien_id' => $pasien['pasien_id'],
            'lokasi_id' => $validatedData['lokasi_id'],
        ]);
    }
}

```

```

        'dokter_id' => $validatedData['dokter_id'],
        'tanggal' => $tanggal,
        'jam_mulai' => $validatedData['jam_mulai'],
        'jam_selesai' => $endTime,
        'status_penginput' => 0
    ]);

    $syncData = [];
    foreach ($validatedData['perawatan_id'] as $perawatanId) {
        $perawatan = Perawatan::find($perawatanId);
        $totalEstimasi += $perawatan->estimasi_waktu_perawatan;
        $syncData[$perawatanId] = [
            'harga' => $perawatan->harga,
            'estimasi_waktu_perawatan' => $perawatan->estimasi_waktu_perawatan,
        ];
    }

    $reservasi_rekam_medi->perawatan()->attach($syncData);

    Alert::success('Success', 'Reservasi berhasil ditambahkan!');
    return redirect()->route('pasien.history');
} catch (Exception $e) {
    dd('simpan data gagal : ', $e);
}
}

public function getBookedTimes(Request $request)
{
    $dokter_id = $request->query('dokter_id');
    $tanggal = $request->query('tanggal');
    $tanggalDate = Carbon::parse($tanggal);

    $bookedTimes = Reservasi::where('dokter_id', $dokter_id)
        ->where('tanggal', $tanggalDate)
        ->get(['jam_mulai', 'jam_selesai']);

    return response()->json($bookedTimes);
}

public function getDokterByLokasi(Request $request)
{
    $lokasi_id = $request->query('lokasi_id');
    $dokter = Dokter::where('lokasi_id', $lokasi_id)->get();
    return response()->json($dokter);
}

public function getAvailableDays(Request $request)
{
    $dokter_id = $request->query('dokter_id');
    $jadwals = DetailJadwal::where('dokter_id', $dokter_id)->pluck('hari');
    return response()->json($jadwals);
}

public function getSesiByDokterAndHari(Request $request)
{
    $dokter_id = $request->query('dokter_id');
    $hari = $request->query('hari');
    $jadwals = DetailJadwal::where('dokter_id', $dokter_id)
        ->where('hari', $hari)
        ->pluck('sesi');
    return response()->json($jadwals);
}

public function destroy($id)
{
    // Hapus entri terkait di tabel detail_jadwal terlebih dahulu
    DB::table('reservasi_rekam_medi')->where('reservasi_id', $id)->delete();

    // Menampilkan alert sukses dan mengarahkan kembali ke halaman data user
    Alert::success('Berhasil!', 'Data reservasi_rekam_medi berhasil dihapus!');
    return redirect('/reservasi/admin');
}

public function terima($id)
{
    $reservasi_rekam_medi = Reservasi::where('reservasi_id', $id)->firstOrFail();

    // Prepare data for reservasi_rekam_medi
}

```

```

$reservasiData = [
    'draft' => 0,
];
// Update the reservasi_rekam_medik
$reservasi_rekam_medik->update($reservasiData);

// Redirect or return a response
Alert::success('Berhasil!', 'Data reservasi_rekam_pasien diterima !');
return redirect()->route('reservasi.index');

}

public function edit($id)
{
    $lokasi = Lokasi::all();
    $perawatan = Perawatan::all();
    $dokter = Dokter::all();
    $reservasi = Reservasi::with(['pasien', 'dokter', 'lokasi'])
        ->where('reservasi_id', $id)
        ->first();

    $tomorrow = Carbon::tomorrow()->format('Y-m-d');

    // Generate available timeslots
    $timeslots = [];
    for ($hour = 10; $hour < 21; $hour++) {
        for ($minute = 0; $minute < 60; $minute += 30) {
            $start = Carbon::createFromTime($hour, $minute);
            $end = $start->copy()->addMinutes(30);
            $timeslots[] = [
                'start' => $start->format('H:i'),
                'end' => $end->format('H:i')
            ];
        }
    }
}

return view('admin.reservasiadmin_edit', compact('lokasi', 'dokter',
    'perawatan', 'reservasi', 'tomorrow'));
}

public function update(Request $request, $id)
{
    try {
        $validatedData = $request->validate([
            'tanggal' => 'required|date|after:today',
            'jam_mulai' => 'required',
            'lokasi_id' => 'required|integer|exists:lokasi,lokasi_id',
            'dokter_id' => 'required|integer',
            'perawatan_id' => 'required|array|max:2',
            'perawatan_id.*' => 'integer|exists:perawatan,perawatan_id',
        ]);

        // Find existing reservation
        $reservasi_rekam_medik = Reservasi::findOrFail($id);

        // Check for time conflicts
        $conflictingReservation = Reservasi::where('tanggal', $validatedData['tanggal'])
            ->where('jam', $validatedData['jam_mulai'])
            ->where('reservasi_id', '!=', $id)
            ->first();

        if ($conflictingReservation) {
            return redirect()->back()->withErrors(['jam_mulai' => 'The selected time is
already booked.'])->withInput();
        }

        // Update reservation data
        $reservasi_rekam_medik->update([
            'lokasi_id' => $validatedData['lokasi_id'],
            'dokter_id' => $validatedData['dokter_id'],
            'status_penginput' => "1",
            'tanggal' => $validatedData['tanggal'],
            'jam' => $validatedData['jam_mulai'],
        ]);
    }

    $reservasi_rekam_medik->perawatan()->sync($validatedData['perawatan_id']);
}

```

```

        $perawatanData = Perawatan::whereIn('perawatan_id',
$validatedData['perawatan_id'])->get();
        $syncData = [];
        foreach ($perawatanData as $perawatan) {
            $syncData[$perawatan->id] = [
                'harga' => $perawatan->harga,
                'estimasi_waktu_perawatan' => $perawatan->estimasi_waktu_perawatan,
            ];
        }

        $reservasi_rekam_medik->perawatan()->update($syncData);

        Alert::success('Berhasil!', 'Data reservasi_rekam_medik pasien berhasil
diperbarui!');
        return redirect()->route('reservasi.index');
    } catch (Exception $e) {
        dd('simpan data gagal : ', $e);
    }
}
}

```

## Kode Controller Lainnya

### 1. ProfileController.php

```

<?php

namespace App\Http\Controllers;

use App\Http\Requests\ProfileUpdateRequest;
use Illuminate\Http\RedirectResponse;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Redirect;
use Illuminate\View\View;

class ProfileController extends Controller
{
    /**
     * Display the user's profile form.
     */
    public function edit(Request $request): View
    {
        return view('profile.edit', [
            'user' => $request->user(),
        ]);
    }

    /**
     * Update the user's profile information.
     */
    public function update(ProfileUpdateRequest $request): RedirectResponse
    {
        $request->user()->fill($request->validated());

        if ($request->user()->isDirty('email')) {
            $request->user()->email_verified_at = null;
        }

        $request->user()->save();

        return Redirect::route('profile.edit')->with('status', 'profile-updated');
    }

    /**
     * Delete the user's account.
     */
    public function destroy(Request $request): RedirectResponse
    {
        $request->validateWithBag('userDeletion', [
            'password' => ['required', 'current_password'],
        ]);
    }
}

```

```

    ]);

    $user = $request->user();

    Auth::logout();

    $user->delete();

    $request->session()->invalidate();
    $request->session()->regenerateToken();

    return Redirect::to('/');
}
}

```

## 2. WhatsAppController.php

```

<?php

namespace App\Http\Controllers;

use Twilio\Rest\Client;
use Illuminate\Http\Request;
use RealRashid\SweetAlert\Facades\Alert;

class WhatsAppController extends Controller
{
    public function sendWhatsAppMessage(Request $request)
    {
        $twilioSid = "AC11c4912c70a2167cd628621406eeefef3";
        $twilioToken = "ffef5f0a4fee7c1bcbb7cd8696d2265d";
        $twilioWhatsAppNumber = "whatsapp:+14155238886";
        $recipientNumber = $request->input('phone'); // Ambil nomor dari request
        $message = $request->input('message'); // Ambil pesan dari request

        $twilio = new Client($twilioSid, $twilioToken);

        try {
            $twilio->messages->create(
                "whatsapp:$recipientNumber",
                [
                    "from" => $twilioWhatsAppNumber,
                    "body" => $message,
                ]
            );

            Alert::success('Success', 'Pesan berhasil dikirim');
            return redirect()->route('reservasi.index');
        } catch (\Exception $e) {
            return response()->json(['error' => $e->getMessage()], 500);
        }
    }
}

```

## Kode Controller Pasien

## 3. ReservasiPasienController.php

```

<?php

namespace App\Http\Controllers\Pasien;

use Session;
use App\Models\Pasien;
use Illuminate\Http\Request;
use App\Models\User;
use App\Models\Lokasi;
use App\Models\Perawatan;
use App\Models\Reservasi;
use App\Models\RekamMedik;
use App\Models\Dokter;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\DB;
use Illuminate\Support\Facades\Hash;
use RealRashid\SweetAlert\Facades\Alert;
use App\Http\Controllers\Controller;
use App\Models\DetailJadwal;
use Illuminate\Support\Carbon;
use Exception;
use Illuminate\Support\Facades\Log;

```

```

use Illuminate\Validation\ValidationException;
use DateTime;
use DateInterval;

class ReservasiPasienController extends Controller
{
    protected $user;

    public function __construct(User $user)
    {
        $this->user = $user;
    }

    public function index()
    {
        $userid = Auth::user();
        $reservasi_rekam_medik = DB::table('reservasi_rekam_medik')
            ->join('pasien', 'reservasi_rekam_medik.pasien_id', '=', 'pasien.pasien_id')
            ->join('lokasi', 'reservasi_rekam_medik.lokasi_id', '=', 'lokasi.lokasi_id')
            ->join("dokter", "reservasi_rekam_medik.dokter_id", "=", "dokter.dokter_id")
            ->where('pasien.user_id', $userid->id)
            ->where('status_penginput', 0)
            ->select('reservasi_rekam_medik.*',
                    'lokasi.*',
                    "pasien.*",
                    'dokter.nama as nama_dokter')->get();

        $reservasi_admin = DB::table("reservasi_rekam_medik")
            ->join("pasien", "reservasi_rekam_medik.pasien_id", "=", "pasien.pasien_id")->get();

        return view('pasien.reservasipasien', compact('reservasi_rekam_medik'));
    }

    public function pasien()
    {
        $reservasi_rekam_medik = DB::table('reservasi_rekam_medik')
            ->join('pasien', 'reservasi_rekam_medik.pasien_id', '=', 'pasien.pasien_id')
            ->join('lokasi', 'reservasi_rekam_medik.lokasi_id', '=', 'lokasi.lokasi_id')
            ->join("dokter", "reservasi_rekam_medik.dokter_id", "=", "dokter.dokter_id")
            ->where('draft', 1)
            ->select('reservasi_rekam_medik.*',
                    'lokasi.*',
                    "pasien.*",
                    'dokter.nama as nama_dokter')->get();

        return view('admin.reservasipasien', compact('reservasi_rekam_medik'));
    }

    public function add()
    {
        $lokasi = Lokasi::all();
        $perawatan = Perawatan::all();
        $dokter = Dokter::all();

        $tomorrow = Carbon::tomorrow()->format('Y-m-d');

        // Generate available timeslots
        $timeslots = [];
        for ($hour = 10; $hour < 21; $hour++) {
            for ($minute = 0; $minute < 60; $minute += 30) {
                $start = Carbon::createFromTime($hour, $minute);
                $end = $start->copy()->addMinutes(30);
                $timeslots[] = [
                    'start' => $start->format('H:i'),
                    'end' => $end->format('H:i')
                ];
            }
        }

        return view('pasien.reservasipasien_add', compact('lokasi', 'dokter', 'perawatan',
            'tomorrow', 'timeslots'));
    }

    public function insert(Request $request)
    {
        try {

```

```

$validatedData = $request->validate([
    'nama' => 'required|string|max:255',
    'tempat_lahir' => 'required|string|max:255',
    'tanggal_lahir' => 'required|date',
    'pekerjaan' => 'required|string|max:255',
    'alamat' => 'required|string',
    'no_Telp' => 'required|string|max:15',
    'lokasi_id' => 'required|integer|exists:lokasi,lokasi_id',
    'dokter_id' => 'required|integer',
    'tanggal' => 'required|date|after:today',
    'jam_mulai' => 'required',
    'perawatan_id' => 'required|array|max:2',
    'perawatan_id.*' => 'integer|exists:perawatan,perawatan_id',
]);
$tanggal = Carbon::parse($validatedData['tanggal'])->format('Y-m-d');

// Calculate total estimated time
$totalEstimasi = 0;
foreach ($validatedData['perawatan_id'] as $perawatanId) {
    $perawatan = Perawatan::find($perawatanId);
    $totalEstimasi += $perawatan->estimasi_waktu_perawatan;
}

$startTime = new DateTime($validatedData['jam_mulai']);
$endTime = (clone $startTime)->modify("+{$totalEstimasi} minutes")->format('H:i');

// Check for time conflicts
$conflictingReservation = Reservasi::where('tanggal', $tanggal)
    ->where(function ($query) use ($validatedData, $endTime) {
        $query->whereBetween('jam_mulai', [$validatedData['jam_mulai'], $endTime])
            ->orWhereBetween('jam_selesai', [$validatedData['jam_mulai'],
$endTime])
            ->orWhereRaw('? BETWEEN jam_mulai AND jam_selesai',
[$validatedData['jam_mulai']])
            ->orWhereRaw('? BETWEEN jam_mulai AND jam_selesai', [$endTime]);
    })
    ->exists();

// Create new reservation
$user_id = Auth::id();

$pasien = Pasien::create([
    'nama' => $validatedData['nama'],
    'tempat_lahir' => $validatedData['tempat_lahir'],
    'tanggal_lahir' => $validatedData['tanggal_lahir'],
    'pekerjaan' => $validatedData['pekerjaan'],
    'alamat' => $validatedData['alamat'],
    'no_Telp' => $validatedData['no_Telp'],
    'user_id' => $user_id,
]);
$reservasi_rekam_medik = Reservasi::create([
    'pasien_id' => $pasien['pasien_id'],
    'lokasi_id' => $validatedData['lokasi_id'],
    'dokter_id' => $validatedData['dokter_id'],
    'tanggal' => $tanggal,
    'jam_mulai' => $validatedData['jam_mulai'],
    'jam_selesai' => $endTime,
    'status_penginput' => 0
]);
$syncData = [];
foreach ($validatedData['perawatan_id'] as $perawatanId) {
    $perawatan = Perawatan::find($perawatanId);
    $totalEstimasi += $perawatan->estimasi_waktu_perawatan;
    $syncData[$perawatanId] = [
        'harga' => $perawatan->harga,
        'estimasi_waktu_perawatan' => $perawatan->estimasi_waktu_perawatan,
    ];
}
$reservasi_rekam_medik->perawatan()->attach($syncData);

Alert::success('Success', 'Reservasi berhasil ditambahkan!');
return redirect()->route('pasien.history');
} catch (Exception $e) {
    dd('simpan data gagal : ', $e);
}

```

```

}

public function getBookedTimes(Request $request)
{
    $dokter_id = $request->query('dokter_id');
    $tanggal = $request->query('tanggal');
    $tanggalDate = Carbon::parse($tanggal);

    $bookedTimes = Reservasi::where('dokter_id', $dokter_id)
        ->where('tanggal', $tanggalDate)
        ->get(['jam_mulai', 'jam_selesai']);

    return response()->json($bookedTimes);
}

public function getDokterByLokasi(Request $request)
{
    $lokasi_id = $request->query('lokasi_id');
    $dokter = Dokter::where('lokasi_id', $lokasi_id)->get();
    return response()->json($dokter);
}

public function getAvailableDays(Request $request)
{
    $dokter_id = $request->query('dokter_id');
    $jadwals = DetailJadwal::where('dokter_id', $dokter_id)->pluck('hari');
    return response()->json($jadwals);
}

public function getSesiByDokterAndHari(Request $request)
{
    $dokter_id = $request->query('dokter_id');
    $hari = $request->query('hari');
    $jadwals = DetailJadwal::where('dokter_id', $dokter_id)
        ->where('hari', $hari)
        ->pluck('sesi');
    return response()->json($jadwals);
}

public function destroy($id)
{
    // Hapus entri terkait di tabel detail_jadwal terlebih dahulu
    DB::table('reservasi_rekam_medik')->where('reservasi_id', $id)->delete();

    // Menampilkan alert sukses dan mengarahkan kembali ke halaman data user
    Alert::success('Berhasil!', 'Data reservasi_rekam_medik berhasil dihapus!');
    return redirect('/reservasi/admin');
}

public function terima($id)
{
    $reservasi_rekam_medik = Reservasi::where('reservasi_id', $id)->firstOrFail();

    // Prepare data for reservasi_rekam_medik
    $reservasiData = [
        'draft' => 0,
    ];
    // Update the reservasi_rekam_medik
    $reservasi_rekam_medik->update($reservasiData);

    // Redirect or return a response
    Alert::success('Berhasil!', 'Data reservasi_rekam_medik pasien diterima !');
    return redirect()->route('reservasi.index');
}

public function edit($id)
{
    $lokasi = Lokasi::all();
    $perawatan = Perawatan::all();
    $dokter = Dokter::all();
    $reservasi = Reservasi::with(['pasien', 'dokter', 'lokasi'])
        ->where('reservasi_id', $id)
        ->first();

    $tomorrow = Carbon::tomorrow()->format('Y-m-d');
}

```

```

// Generate available timeslots
$timeslots = [];
for ($hour = 10; $hour < 21; $hour++) {
    for ($minute = 0; $minute < 60; $minute += 30) {
        $start = Carbon::createFromTime($hour, $minute);
        $end = $start->copy()->addMinutes(30);
        $timeslots[] = [
            'start' => $start->format('H:i'),
            'end' => $end->format('H:i')
        ];
    }
}

return view('admin.reservasiadmin_edit', compact('lokasi', 'dokter',
'perawatan','reservasi', 'tomorrow'));
}

public function update(Request $request, $id)
{
    try {
        $validatedData = $request->validate([
            'tanggal' => 'required|date|after:today',
            'jam_mulai' => 'required',
            'lokasi_id' => 'required|integer|exists:lokasi,lokasi_id',
            'dokter_id' => 'required|integer',
            'perawatan_id' => 'required|array|max:2',
            'perawatan_id.*' => 'integer|exists:perawatan,perawatan_id',
        ]);
    }

    // Find existing reservation
    $reservasi_rekam_medik = Reservasi::findOrFail($id);

    // Check for time conflicts
    $conflictingReservation = Reservasi::where('tanggal', $validatedData['tanggal'])
        ->where('jam', $validatedData['jam_mulai'])
        ->where('reservasi_id', '!=', $id)
        ->first();

    if ($conflictingReservation) {
        return redirect()->back()->withErrors(['jam_mulai' => 'The selected time is
already booked.']);
    }

    // Update reservation data
    $reservasi_rekam_medik->update([
        'lokasi_id' => $validatedData['lokasi_id'],
        'dokter_id' => $validatedData['dokter_id'],
        'status_penginput' => "1",
        'tanggal' => $validatedData['tanggal'],
        'jam' => $validatedData['jam_mulai'],
    ]);

    $reservasi_rekam_medik->perawatan()->sync($validatedData['perawatan_id']);
    $perawatanData = Perawatan::whereIn('perawatan_id',
$validatedData['perawatan_id'])->get();
    $syncData = [];
    foreach ($perawatanData as $perawatan) {
        $syncData[$perawatan->id] = [
            'harga' => $perawatan->harga,
            'estimasi_waktu_perawatan' => $perawatan->estimasi_waktu_perawatan,
        ];
    }

    $reservasi_rekam_medik->perawatan()->update($syncData);

    Alert::success('Berhasil!', 'Data reservasi_rekam_medik pasien berhasil
diperbarui!');
    return redirect()->route('reservasi.index');
} catch (Exception $e) {
    dd('simpan data gagal : ', $e);
}
}
}

```

## Kode Models

### 1. Admin.php

```

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Admin extends Model
{
    use HasFactory;

    protected $table = 'admin';
    protected $primaryKey = 'admin_id';
    public $timestamps = false;

    protected $fillable = [
        'nama',
        'alamat',
        'email',
        'password'
    ];

    public function reservasi()
    {
        return $this->hasMany(Reservasi::class, 'admin_id');
    }

    public function pengeluaran()
    {
        return $this->hasMany(Pengeluaran::class, 'admin_id');
    }
}

```

## 2. DetailDokter.php

```

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class DataDokter extends Model
{
    protected $table = 'dokter';

    public function users()
    {
        return $this->hasOne(User::class, 'id', 'id');
    }

    public function allData()
    {
        return DB::table('dokter')->get();
    }

    public function addData($data)
    {
        DB::table('dokter')->insert($data);
    }
}

```

## 3. DetailJadwal.php

```

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class DetailJadwal extends Model
{
    use HasFactory;

    protected $table = 'detail_jadwal';
    protected $primaryKey = 'jadwal_id';
    public $timestamps = false;
}

```

```

protected $fillable = [
    'dokter_id',
    'hari',
    'sesi'
];
public function dokter()
{
    return $this->belongsTo(Dokter::class, 'dokter_id');
}
public function allData()
{
    return DB::table('detail_jadwal')->get();
}
public function addData($data)
{
    DB::table('detail_jadwal')->insert($data);
}

```

#### 4. Dokter.php

```

<?php
namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Dokter extends Model
{
    use HasFactory;

    protected $table = 'dokter';
    protected $primaryKey = 'dokter_id';
    public $timestamps = false;

    protected $fillable = [
        'nama',
        'alamat',
        'nomor_hp',
        'lokasi_id',
        'user_id',
    ];
    public function detailJadwal()
    {
        return $this->hasMany(DetailJadwal::class, 'dokter_id');
    }
    public function izin()
    {
        return $this->hasMany(Izin::class, 'dokter_id');
    }
    public function rekamMedik()
    {
        return $this->hasMany(RekamMedik::class, 'dokter_id');
    }
}

```

#### 5. Izin.php

```

<?php
namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class Izin extends Model
{
    use HasFactory;

    protected $table = 'izin_dokter';
    protected $primaryKey = 'izin_id';
    public $timestamps = false;

    protected $fillable = [

```

```

        'user_id',
        'tanggal_awal',
        'tanggal_akhir',
        'alasan',
        'status',
    ];

    public function allData()
    {
        return DB::table('izin_dokter')->get();
    }

    public function addData($data)
    {
        DB::table('izin_dokter')->insert($data);
    }

    public function users()
    {
        return $this->belongsTo(User::class, 'user_id');
    }
}

```

## 6. Lokasi.php

```

<?php
namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

class Lokasi extends Model
{
    use HasFactory;

    protected $table = 'lokasi';
    protected $primaryKey = 'lokasi_id';
    public $timestamps = false;

    protected $fillable = [
        'nama_lokasi',
        'alamat'
    ];

    public function reservasi()
    {
        return $this->hasMany(Reservasi::class, 'lokasi_id');
    }
}

```

## 7. Pasien.php

```

<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class Pasien extends Model
{
    protected $table = 'pasien';
    protected $primaryKey = 'pasien_id';
    public $timestamps = false;

    protected $fillable = [
        'nama',
        'tempat_lahir',
        'tanggal_lahir',
        'pekerjaan',
        'alamat',
        'no_Telp',
        'user_id'
    ];

    public function users()
    {
        return $this->hasOne(User::class, 'id', 'id');
    }
}

```

```

public function allData()
{
    return DB::table('pasien')->get();
}

public function addData($data)
{
    DB::table('pasien')->insert($data);
}

public function pasien()
{
    return $this->hasMany(Reservasi::class, 'id', 'id');
}
}

```

## 8. Pengeluaran.php

```

<?php
namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class Pengeluaran extends Model
{
    use HasFactory;

    protected $table = 'pengeluaran';
    protected $primaryKey = 'pengeluaran_id';
    public $timestamps = false;

    protected $fillable = [
        'admin_id',
        'deskripsi_pengeluaran',
        'nama_pengeluaran',
        'kategori_pengeluaran',
        'jumlah_pengeluaran',
        'tanggal'
    ];

    public function admin()
    {
        return $this->belongsTo(Admin::class, 'admin_id');
    }

    public function allData()
    {
        return DB::table('pengeluaran')->get();
    }

    public function addData($data)
    {
        DB::table('pengeluaran')->insert($data);
    }
}

```

## 9. Perawatan.php

```

<?php
namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class Perawatan extends Model
{
    use HasFactory;

    protected $table = 'perawatan';
    protected $primaryKey = 'perawatan_id';
    public $timestamps = false;

    protected $fillable = [
        'jenis_Perawatan',
        'harga',
        'estimasi_waktu_perawatan'
    ];
}

```

```

public function reservasi()
{
    return $this->belongsToMany(Reservasi::class, 'perawatan_reservasi', 'perawatan_id',
'reservasi_id');
}

public function allData()
{
    return DB::table('perawatan')->get();
}

public function addData($data)
{
    DB::table('perawatan')->insert($data);
}

```

## 10. PerawatanReservasi.php

```

<?php
namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class PerawatanReservasi extends Model
{
    use HasFactory;

    protected $table = 'perawatan_reservasi';
    protected $primaryKey = 'id';
    public $timestamps = false;

    protected $fillable = [
        'perawatan_id',
        'reservasi_id',
        'harga',
        'estimasi_waktu_perawatan'
    ];

    public function reservasi()
    {
        return $this->belongsTo(Reservasi::class, 'reservasi_id', 'reservasi_id');
    }

    public function perawatan()
    {
        return $this->belongsTo(Perawatan::class, 'perawatan_id', 'perawatan_id');
    }

    public function allData()
    {
        return DB::table('perawatan_reservasi')->get();
    }
}

```

## 11. RekamMedik.php

```

<?php
namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class RekamMedik extends Model
{
    use HasFactory;

    protected $table = 'rekam_medic';
    protected $primaryKey = 'rekam_id';
    public $timestamps = false;

    protected $fillable = [
        'reservasi_id',
    ];
}

```

```

        'dokter_id',
        'perawatan_id',
        'golongan_darah',
        'tekanan_darah',
        'penyakit_jantung',
        'diabetes',
        'hepatitis',
        'penyakit_lainnya',
        'alergi_makanan',
        'alergi_obat',
        'keluhan',
        'gigi',
        'biaya'
    ];

    public function reservasi()
    {
        return $this->belongsTo(Reservasi::class, 'reservasi_id', 'reservasi_id');
    }

    public function dokter()
    {
        return $this->belongsTo(Dokter::class, 'dokter_id');
    }

    public function allData()
    {
        return DB::table('rekam_medik')->get();
    }

    public function addData($data)
    {
        DB::table('rekam_medik')->insert($data);
    }
}

```

## 12. Reservasi.php

```

<?php
namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;
use Illuminate\Support\Facades\DB;

class Reservasi extends Model
{
    use HasFactory;

    protected $table = 'reservasi_rekam_medik';
    protected $primaryKey = 'reservasi_id';
    public $timestamps = false;

    protected $fillable = [
        'pasien_id',
        'lokasi_id',
        'dokter_id',
        'admin_id',
        'rekam_medik_id',
        'tanggal',
        'jam_mulai',
        'jam_selesai',
        'golongan_darah',
        'tekanan_darah',
        'penyakit_jantung',
        'diabetes',
        'hepatitis',
        'penyakit_lainnya',
        'alergi_makanan',
        'alergi_obat',
        'keluhan',
        'gigi',
        'draft',
        'status_penginput',
    ];

    public function allData()
    {
        return DB::table('reservasi_rekam_medik')->get();
    }
}

```

```

public function pasien()
{
    return $this->belongsTo(Pasien::class, 'pasien_id');
}

public function dokter()
{
    return $this->belongsTo(Dokter::class, 'dokter_id');
}

public function lokasi()
{
    return $this->belongsTo(Lokasi::class, 'lokasi_id');
}

public function perawatan()
{
    return $this->belongsToMany(Perawatan::class, 'perawatan_reservasi', 'reservasi_id',
'perawatan_id')
        ->withPivot('harga', 'estimasi_waktu_perawatan');
}

// Metode untuk menghitung total harga perawatan
public function getTotalHargaPerawatanAttribute()
{
    return $this->perawatan->sum('pivot.harga');
}

public function user()
{
    return $this->belongsTo(Admin::class, 'id');
}

public function rekamMedik()
{
    return $this->hasOne(RekamMedik::class, 'reservasi_id', 'reservasi_id');
}

public function perawatan_reservasi()
{
    return $this->hasMany(PerawatanReservasi::class, 'reservasi_id', 'reservasi_id');
}

}

```

### 13. User.php

```

<?php

namespace App\Models;

// use Illuminate\Contracts\Auth\MustVerifyEmail;
use Illuminate\Support\Facades\DB;
use Illuminate\Notifications\Notifiable;
use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Foundation\Auth\User as Authenticatable;

class User extends Authenticatable
{
    use HasFactory, Notifiable;

    /**
     * The attributes that are mass assignable.
     *
     * @var array<int, string>
     */
    protected $fillable = [
        'role',
        'name',
        'email',
        'password',
    ];

    /**
     * The attributes that should be hidden for serialization.
     *
     * @var array<int, string>
     */
    protected $hidden = [

```

```
        'password',
        'remember_token',
    ];

/**
 * Get the attributes that should be cast.
 *
 * @return array<string, string>
 */
protected function casts(): array
{
    return [
        'email_verified_at' => 'datetime',
        'password' => 'hashed',
    ];
}

public function allData()
{
    return DB::table('users')->get();
}

public function addData($data)
{
    $user = User::create($data);
    return $user->id;
}
```

**LAMPIRAN F**  
**PENGUJIAN SISTEM**  
**(LANJUTAN)**

## 1. Admin

### LEMBAR PENGUJIAN APLIKASI

Nama Aplikasi : Sistem Informasi Manajemen Klinik Gigi

Nama Penguji : *Fath Nahr Febrina*

Hak Akses/Jabatan : Admin

No	Fungsional	Skenario	Hasil yang diharapkan	Hasil	
				Sesuai	Tidak Sesuai
1	Authentikasi dan Pengelolaan akun	Login	Sistem berhasil login dan menampilkan dashboard	✓	
		Ganti Password	Sistem berhasil mengubah password akun	✓	
		Edit Profil	Sistem berhasil mengubah profil akun	✓	
		Hapus Akun	Sistem berhasil menghapus akun user	✓	
2	Lihat Dasboard	Lihat	Sistem berhasil menampilkan dashboard	✓	
		Filter	Sistem berhasil menampilkan informasi yang berbeda dari dashborad	✓	
3	Konfirmasi reservasi dari Pasien	Lihat	Sistem berhasil menampilkan list reservasi yang diajukan pasien	✓	
		Tolak	Sistem berhasil menolak reservasi dari pasien	✓	
		Terima	Sistem berhasil menerima reservasi dari pasien.	✓	
4	Reservasi pasien	Lihat	Sistem berhasil menampilkan list reservasi yang dibuat admin	✓	
		Tambah	Sistem berhasil menambahkan reservasi	✓	
		Edit	Sistem berhasil mengedit informasi reservasi	✓	
5	Mengelola data pasien	Lihat	Sistem berhasil menampilkan list pasien	✓	
		Edit	Sistem berhasil mengedit informasi pasien	✓	
		Hapus	Sistem berhasil menghapus data pasien	✓	
6	Mengelola data	Lihat	Sistem berhasil	✓	

	rekam medik		menampilkan rekam medik pasien	✓	
		Edit	Sistem berhasil mengedit informasi rekam medik pasien	✓	
7	Mengelola data dokter, jadwal dokter	Lihat	Sistem berhasil menampilkan data dokter dan jadwal praktek dokter	✓	
		Tambah	Sistem berhasil menambahkan data dokter dan jadwal praktek dokter	✓	
		Edit	Sistem berhasil mengedit data dokter dan jadwal praktek dokter	✓	
		Hapus	Sistem berhasil menghapus data dokter dan jadwal praktek dokter	✓	
8	Mengelola data user	Lihat	Sistem informasi berhasil menampilkan list user web	✓	
		Tambah	Sistem informasi berhasil menambahkan user	✓	
		Edit	Sistem informasi berhasil mengedit user	✓	
		Hapus	Sistem informasi berhasil menghapus user	✓	

Padang, September 2024

Penguji



( Falen Nida F. )

## 2. Owner

Nama Aplikasi : Sistem Informasi Manajemen Klinik Gigi

Nama Penguji : Ibg. Muhammad Patri Ramadhan

Hak Akses/Jabatan : Owner

No	Fungsional	Skenario	Hasil yang diharapkan	Hasil	
				Sesuai	Tidak Sesuai
1	Authentikasi dan Pengelolaan akun	Login	Sistem berhasil login dan menampilkan dashboard	✓	
		Ganti Password	Sistem berhasil mengubah password akun	✓	
		Edit Profil	Sistem berhasil mengubah profil akun	✓	
		Hapus Akun	Sistem berhasil menghapus akun user	✓	
2	Lihat Dasboard	Lihat	Sistem berhasil menampilkan dashboard	✓	
		Filter	Sistem berhasil menampilkan informasi yang berbeda dari dashborad	✓	
3	Melihat data profit klinik	Lihat	Sistem berhasil menampilkan profit klinik	✓	
		Filter	Sistem berhasil mengfilter profit berdasarkan bulan	✓	
4	Melihat jumlah kunjungan pasien	Lihat	Sistem berhasil menampilkan jumlah kunjungan pasien	✓	
		Filter	Sistem berhasil mengfilter jumlah kunjungan berdasarkan bulan	✓	
5	Melihat jumlah penanganan perawatan berdasarkan kategori	Lihat	Sistem berhasil menampilkan jumlah penanganan perawatan berdasarkan kategori	✓	
		Filter	Sistem berhasil mengfilter jumlah penanganan perawatan berdasarkan bulan	✓	
6	Melihat performa dokter berdasarkan jumlah pasien	Lihat	Sistem berhasil menampilkan data peforma dokter berdasarkan jumlah pasien yang dirawat	✓	
		Filter	Sistem berhasil mengfilter data jumlah pasien yang dirawat dokter	✓	
7	Melihat data	Lihat	Sistem berhasil menampilkan	✓	

	pasien berdasarkan umur		data pasien berdasarkan umurnya	✓	
		Filter	Sistem berhasil mengfilter profit berdasarkan bulan	✓	
8	Melihat Data dokter	Lihat	Sistem berhasil menampilkan list data dokter	✓	
9	Melihat Data Pasien	Lihat	Sistem berhasil menampilkan list data pasien	✓	
10	Melihat Data Pemasukan	Lihat	Sistem berhasil menampilkan list data pemasukan	✓	
11	Memutuskan perizinan dokter	Terima	Sistem berhasil menerima perizinan dokter	✓	
		Tolak	Sistem berhasil menolak perizinan dokter	✓	
12	Mengelola pencatatan pengeluaran klinik	Lihat	Sistem berhasil menampilkan pengeluaran klinik	✓	
		Tambah	Sistem berhasil menambah pengeluaran klinik	✓	
		Edit	Sistem berhasil mengedit pengeluaran klinik	✓	
		Hapus	Sistem berhasil menghapus pengeluaran klinik	✓	
13	Mengelola harga perawatan	Lihat	Sistem berhasil menampilkan harga perawatan	✓	
		Tambah	Sistem berhasil menambahkan harga perawatan	✓	
		Edit	Sistem berhasil mengedit harga perawatan	✓	
		Hapus	Sistem berhasil menghapus harga perawatan		

Padang, September 2024

Penguji

(drg. Muhammad Atri Rafadhan)

### 3. Dokter

Nama Aplikasi : Sistem Informasi Manajemen Klinik Gigi

Nama Penguji : Drg. Farhan Muhammad Nave

Hak Akses/Jabatan : Dokter

No	Fungsional	Skenario	Hasil yang diharapkan	Hasil	
				Sesuai	Tidak Sesuai
1	Authentikasi dan Pengelolaan akun	Login	Sistem berhasil login dan menampilkan dashboard	✓	
		Ganti Password	Sistem berhasil mengubah password akun	✓	
		Edit Profil	Sistem berhasil mengubah profil akun	✓	
		Hapus Akun	Sistem berhasil menghapus akun user	✓	
2	Lihat Jadwal Praktek	Lihat	Sistem berhasil menampilkan jadwal praktek dokter	✓	
3	Mengelola data pasien	Lihat	Sistem berhasil menampilkan list data pasien	✓	
		Edit	Sistem berhasil mengedit list data pasien	✓	
		Hapus	Sistem berhasil menghapus list data pasien	✓	
4	Mengajukan permintaan izin	Lihat	Sistem berhasil menampilkan list permintaan izin yang diajukan	✓	
		Tambah	Sistem berhasil mengajukan permintaan izin ke owner	✓	
		Edit	Sistem berhasil mengedit permintaan izin ke owner	✓	
		Hapus	Sistem berhasil menghapus permintaan izin ke owner	✓	

Padang, September 2024

Penguji

(Drg.Farhan Muhammad Nave )

#### 4. Pasien

Nama Aplikasi : Sistem Informasi Manajemen Klinik Gigi

Nama Penguji : Laras Adinda Putry

Hak Akses/Jabatan : Pasien

No	Fungsional	Skenario	Hasil yang diharapkan	Hasil	
				Sesuai	Tidak Sesuai
1	Authentikasi dan Pengelolaan akun	Login	Sistem berhasil login dan menampilkan dashboard	✓	
		Ganti Password	Sistem berhasil mengubah password akun	✓	
		Edit Profil	Sistem berhasil mengubah profil akun	✓	
		Hapus Akun	Sistem berhasil menghapus akun user	✓	
2	Registrasi Akun	Registrasi	Sistem berhasil melakukan registrasi akun	✓	
3	Reservasi klinik	Tambah	Sistem berhasil mengajukan reservasi ke klinik	✓	
4	Melihat Riwayat reservasi	Lihat	Sistem berhasil melihat riwayat reservasi yang sudah pernah diajukan	✓	

Padang, September 2024

Penguji



(LARAS ADINDA PUTRY ,)

**LAMPIRAN G**

**DATA DOKUMEN**

## 1. Berita Acara Wawancara

### Berita Acara Wawancara

Waktu wawancara : 02 Oktober 2023

Lokasi Wawancara : Cafe Kopi Roastery, Padang

#### Profil Pewawancara

Nama : Alif Abdul Rauf

NIM : 2011522024

#### Profil Narasumber

Nama : Arsy. Muhammad Rizki Ramadhan

Jabatan : Owner

#### Hasil Wawancara

Pertanyaan Bagaimana proses pendaftaran untuk bisa melakukan perawatan pada klinik xenon dental house?

Jawaban Saat ini pendaftaran untuk perawatan bisa dilakukan dengan 2 cara. Pertama pasien bisa menghubungi klinik melalui no whatsapp klinik gigi xenon dental house. Pasien akan diminta untuk mengisi format pesan yang sudah disediakan dari klinik, jika sudah dikirim dan telah sesuai maka pasien akan terdaftar untuk reservasinya. Kedua pasien bisa langsung mendatangi klinik untuk melakukan reservasi. Untuk reservasi ini akan dibantu dengan admin dari klinik ini secara langsung.

Pertanyaan Proses apa yang terjadi setelah dilakukannya reservasi?

Jawaban Pada hari perawatan, admin akan mengingatkan pasien 1 jam sebelum waktu perawatan untuk datang ke klinik sesuai dengan waktu yang sudah direservasi. Saat pasien sudah datang, admin akan mengisikan data pribadi pasien pada lembar awal rekam medik. Setelah itu pasien diarahkan masuk ke ruangan perawatan untuk dilakukan perawatan oleh dokter. Dokter akan memeriksa

pasien, bertanya kepada pasien terkait kondisi pasien, riwayat penyakit dan dokter akan mengisikan rekam medik pasien. Setelah itu akan dilakukan perawatan pasien, perawatan ini bisa berupa cabut gigi, behel, tambal gigi, *scaling*, *bleaching*, gigi tiruan. Setelah selesai admin akan memeriksa rekam medik pasien dan meminta pasien untuk melakukan pembayaran. Jika dari proses perawatan reservasi yang dilakukan memerlukan perawatan lebih lanjut maka pasien akan direservasikan ulang untuk perawatan selanjutnya.

Pertanyaan	Apa saja peran yang ada pada klinik xenon dental house?
Jawaban	Di klinik ini terbagi menjadi 2 yaitu ada bagian administrasi dan tenaga medis. Untuk administrasi adalah orang yang bertanggung jawab mengelola aspek administratif klinik, seperti pengelolaan jadwal, sosial media, dan resepsionis. Untuk tenaga medis terdiri dari dokter dan asisten dokter yang akan melakukan perawatan kepada pasien. Pada klinik ini juga terdapat dokter pengganti yang akan menggantikan dokter tetap di klinik ini jika dokter tetap sedang izin untuk tidak bekerja kepada owner.
Pertanyaan	Apa saja peran yang ada pada klinik xenon dental house?
Jawaban	Di klinik ini terbagi menjadi 2 yaitu ada bagian administrasi dan tenaga medis. Untuk administrasi adalah orang yang bertanggung jawab mengelola aspek administratif klinik, seperti pengelolaan jadwal, sosial media, dan resepsionis. Untuk tenaga medis terdiri dari dokter dan asisten dokter yang akan melakukan perawatan kepada pasien. Pada klinik ini juga terdapat dokter pengganti yang akan menggantikan dokter tetap di klinik ini jika dokter tetap sedang izin untuk tidak bekerja kepada owner.
Pertanyaan	Dari proses bisnis yang ada ini apakah ada fitur tambahan yang dirasa dibutuhkan untuk klinik xenon dental house?
Jawaban	Dibutuhkan rekapan atau ringkasan data yang berisi data kategori penanganan, pasien dan keuangan. Sehingga data yang bersangkutan bisa dipantau dan bisa dijadikan data untuk membantu dalam pengambilan keputusan.

Padang, Oktober 2023

Narasumber

(Dr. Muhammad Atri) Ramadhan

## 2. Surat Pernyataan Owner

### SURAT PERNYATAAN

Yang bertanda tangan dibawah ini :

Nama : drg. Muhammad Afri Ramadhan

Jabatan : Owner

Menyatakan bahwa mahasiswa Perguruan Tinggi Negeri Berbadan Hukum yang bernama :

Nama : Alif Abdul Rauf

NIM : 2011522024

Departemen/Fakutas : Sistem Informasi/Teknologi Informasi

PTN BH : Universitas Andalas

Terkait Tugas Akhir “Rancang Bangun Sistem Informasi Manajemen Klinik Gigi Studi Kasus : Klinik Gigi Xenon Dental House” bahwa aplikasi yang dibangun telah sesuai dengan bisnis proses yang ada dan sesuai kebutuhan klinik gigi xenon dental house departemen Sistem Informasi.

Demikian surat pernyataan ini saya berikan untuk digunakan dengan sebagaimana mestinya.

Padang, September 2024

Pengaji



(drg. Muhammad Afri Ramadhan)

### 3. Surat Pernyataan Admin

#### **SURAT PERNYATAAN**

Yang bertanda tangan dibawah ini :

Nama : Falen Nela Febrianti  
Jabatan : Admin

Menyatakan bahwa mahasiswa Perguruan Tinggi Negeri Berbadan Hukum yang bernama :

Nama : Alif Abdul Rauf  
NIM : 2011522024  
Departemen/Fakutas : Sistem Informasi/Teknologi Informasi  
PTN BH : Universitas Andalas

Terkait Tugas Akhir “Rancang Bangun Sistem Informasi Manajemen Klinik Gigi Studi Kasus : Klinik Gigi Xenon Dental House” bahwa aplikasi yang dibangun telah sesuai dengan bisnis proses yang ada dan sesuai kebutuhan klinik gigi xenon dental house departemen Sistem Informasi.

Demikian surat pernyataan ini saya berikan untuk digunakan dengan sebagaimana mestinya.

Padang, September 2024

Penguji



( Falen Nela Febrianti )

#### 4. Surat Pernyataan Dokter

#### **SURAT PERNYATAAN**

Yang bertanda tangan dibawah ini :

Nama : Drg. Farhan Muhamad Nouv<sup>z</sup>  
Jabatan : Dokter

Menyatakan bahwa mahasiswa Perguruan Tinggi Negeri Berbadan Hukum yang bernama :

Nama : Alif Abdul Rauf  
NIM : 2011522024  
Departemen/Fakutas : Sistem Informasi/Teknologi Informasi  
PTN BH : Universitas Andalas

Terkait Tugas Akhir “Rancang Bangun Sistem Informasi Manajemen Klinik Gigi Studi Kasus : Klinik Gigi Xenon Dental House” bahwa aplikasi yang dibangun telah sesuai dengan bisnis proses yang ada dan sesuai kebutuhan klinik gigi xenon dental house departemen Sistem Informasi.

Demikian surat pernyataan ini saya berikan untuk digunakan dengan sebagaimana mestinya.

Padang, September 2024

Penguji



(Drg. Farhan Muhamad Nouv)

## 5. Surat Pernyataan Pasien

### **SURAT PERNYATAAN**

Yang bertanda tangan dibawah ini :

Nama : Laras Adinda Putri  
Jabatan : Pasien

Menyatakan bahwa mahasiswa Perguruan Tinggi Negeri Berbadan Hukum yang bernama :

Nama : Alif Abdul Rauf  
NIM : 2011522024  
Departemen/Fakutas : Sistem Informasi/Teknologi Informasi  
PTN BH : Universitas Andalas

Terkait Tugas Akhir “Rancang Bangun Sistem Informasi Manajemen Klinik Gigi Studi Kasus : Klinik Gigi Xenon Dental House” bahwa aplikasi yang dibangun telah sesuai dengan bisnis proses yang ada dan sesuai kebutuhan klinik gigi xenon dental house departemen Sistem Informasi.

Demikian surat pernyataan ini saya berikan untuk digunakan dengan sebagaimana mestinya.

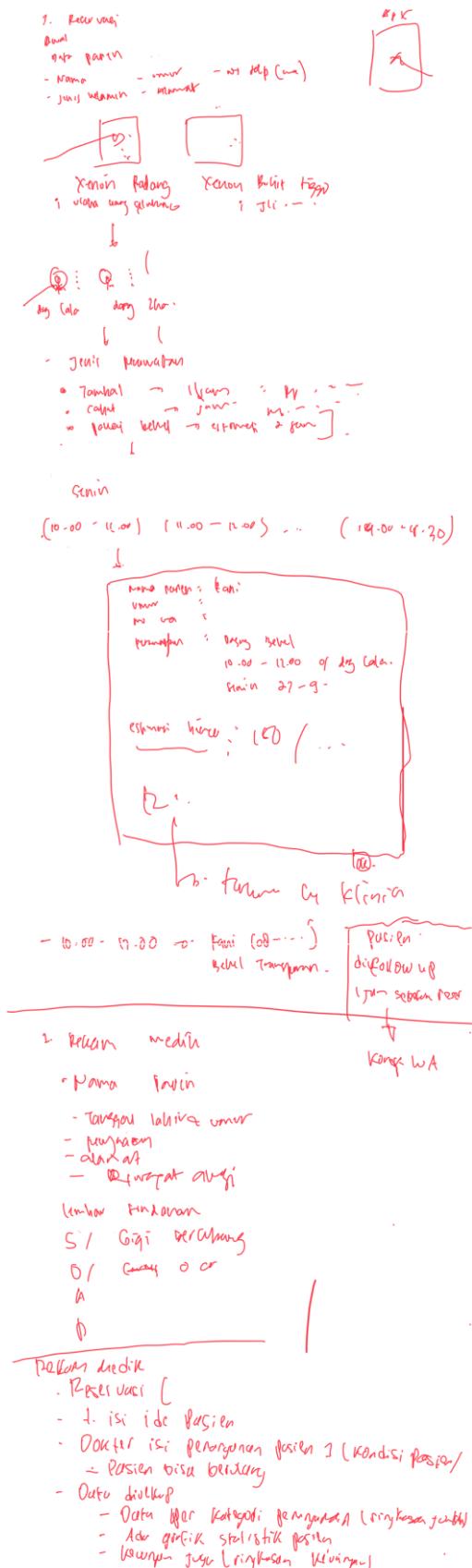
Padang, September 2024

Penguji



(LARAS ADINDA PUTRI )

## 6. Bukti hasil brainstorming



## 7. Buku Pengeluaran

**MEI**

Tanggal / Date	KETERANGAN / DESCRIPTION	Debet / Debit	Kredit / Credit
3 5	Handtissue 5 (2) Isoture 0.25 (1) Medental gold (1) ( MASUK Bon Bg Farhan ) ~ GLOBAL DENT ~		
3 5	A PRODENTAL Seloid Strip @ CN Slicing Paper Power 0	75.000 45.000 60.000	
4 5	RK Palfrise Lx5 OTO A PRODENTAL.	RP 780.000	
		TOTAL = RP 395.000	
14 5	Elastis Medium (3) - " SOFT (3) " " GLOBAL DENT " Handtissue (3)	RP 204.000	[LUNAS]
14 5	Saliva (2) " A PRODENTAL "	TOTAL RP 86.000	
15 5	Ultrasonic Scaler D3 LED Selang 4x6 mm SMC Neple T (3 joint)	RP 2.600.000	[GLOBAL DENT]
		RP 77.000	" GLOBAL DENT "
		Total / Sub Total *	

## 8. Excel Pemasukan

The screenshot shows an Excel spreadsheet titled "LAPORAN HARIAN PENDAPATAN XENON DENTAL HOUSE Minggu/ 26 Mei 2024". The table has columns for No, NAMA PASIEN, GIG, NAMA PERAWATA, BAHAN TERPAKA, BIAYA, BAYAR, SISA, DOKTER, and KETERANGAN. Rows 7, 8, and 9 show entries for patients 1, 2, and 3 respectively. Row 15 is a total row. Below the table, notes mention Dokter Pagi (Org. Atul), Dokter Siang (Drg. Atul), Asop (Viona), and Admin (Melda). The status bar at the bottom shows the date as 26 mei 2024.

No	NAMA PASIEN	GIG	NAMA PERAWATA	BAHAN TERPAKA	BIAYA	BAYAR	SISA	DOKTER	KETERANGAN
1	[REDACTED]			retainer	Rp 500.000	Rp 500.000		drg atul	tf
2	[REDACTED]			kontrol	Rp 220.000	Rp 220.000		drg atul	tf
3									
Total					Rp 720.000	Rp 720.000			
18	Dokter Pagi			Org. Atul					
19	Dokter Siang			Drg. Atul					
20	Asop			: Viona					
21	Admin			: Melda					

## 9. Rekam Medik

  
**XENON DENTAL HOUSE**  
 (CLINIC & STORE)  
 JL. Palembang No.11 Uluh Kuning Selatan, Kec. Padang Utara, Kota Padang  
 Telp. 0823-8715-3853 / IG :@xendentalhouse

REKAM MEDIK PASIEN	
NO. RM	Tanggal
DATA PASIEN :	
Nama: [REDACTED] Tempat/Tanggal Lahir: [REDACTED] Pekerjaan: [REDACTED] Alamat Rumah: [REDACTED] No. HP / WA: [REDACTED]	
DATA MEDIK	
Golongan Darah:	A / B / AB / O
Tekanan Darah:	
Penyakit Jantung:	Tidak ada / ada :
Diabetes:	Tidak ada / ada :
Hepatitis:	Tidak ada / ada :
Penyakit Lainnya:	Tidak ada / ada :
Allergi Obat:	Tidak ada / ada :
Makanan:	Tidak ada / ada :

  
**XENON DENTAL HOUSE**  
 (CLINIC & STORE)  
 JL. Palembang No.11 Uluh Kuning Selatan, Kec. Padang Utara, Kota Padang  
 Telp. 0823-8715-3853 / IG :@xendentalhouse

TANGGAL	GIGI	KELUHAN/ DIAGNOSA	PERAWATAN	BIAYA	DOKTER
20 Nov 2018	51	1/ begin akar 2/ begin akar a/ tukup (+) poros c/ fusi apik (+) begin J (+) O- car need b/ fusi apik (+) sengkel	tr/ p/p Capping + RE-Klas I	Rp. 300.000	J.