

BAB IV HASIL DAN PEMBAHASAN

4.1. Hasil Implementasi Sistem

Bagian ini menguraikan hasil implementasi sistem yang telah dibangun menggunakan *framework Laravel* dan *Framework Laravel* difungsikan sebagai penghubung antara layanan *home care*. Anda bisa menjelaskan bagaimana *Laravel* digunakan untuk membangun sistem berbasis *web* yang menampilkan data layanan dan pemesanan untuk pelanggan.

Contoh Penjelasan:

1. Penggunaan *Laravel*:

Laravel digunakan untuk membangun sistem berbasis *web* yang menghubungkan antara *frontend* dan *backend* untuk menampilkan layanan, pemesanan, dan status kegiatan.

2. Tampilan *Web*:

Tampilkan *screenshot website* yang menggunakan data dan diakses melalui *Laravel*. Jelaskan bagaimana data ditampilkan secara dinamis dengan menggunakan *Blade templates* di *Laravel*.

3. Pengolahan Data:

Jelaskan bagaimana disusun untuk memetakan layanan yang tersedia dan bagaimana data diakses serta disajikan di.

4. Database *MySQL*:

Bahas penggunaan *MySQL* untuk menyimpan data terkait layanan *home care*, serta bagaimana *Laravel* berfungsi untuk mengelola data tersebut dengan *Eloquent ORM*.

Sistem yang dikembangkan menggunakan *Laravel* sebagai *backend* dan untuk manajemen data layanan. Dalam *frontend*, menggunakan *Blade templates* untuk menampilkan data layanan secara dinamis. Setiap layanan yang dicari oleh pelanggan dikategorikan berdasarkan area rumah dan jenis pekerjaan, yang kemudian diambil dari melalui *query SPARQL*. Data ini diproses dan ditampilkan sesuai dengan format yang diinginkan oleh pelanggan.

Buat Akun Baru

Bergabung dan nikmati layanan homecare terbaik

Nama Lengkap

Email

Password

Konfirmasi Password

Daftar Sekarang

Sudah punya akun? [Masuk](#)

Gambar 4. 1 Halaman *Login*

Tampilan Halaman Registrasi Pengguna Pada halaman "Buat Akun Baru" yang ada di aplikasi *HomeCare*, pengguna dapat melakukan pendaftaran untuk membuat akun agar dapat memanfaatkan layanan *home care*. Berikut adalah elemen-elemen yang ada pada halaman registrasi:

1. *Form Input* Pengguna:

Formulir ini meminta pengguna untuk mengisi data berikut:

2. Nama Lengkap:

Pengguna diminta untuk mengisi nama lengkap mereka, yang akan digunakan untuk mempersonalisasi akun mereka di aplikasi.

3. *Email:*

Pengguna memasukkan alamat *email* yang akan digunakan untuk verifikasi dan komunikasi lebih lanjut.

4. *Password:*

Pengguna membuat kata sandi untuk mengamankan akun mereka.

5. Konfirmasi *Password:*

Pengguna diminta untuk mengetikkan ulang kata sandi yang telah dibuat untuk memastikan tidak ada kesalahan penulisan.

6. Tombol Daftar Sekarang:

Setelah mengisi formulir, pengguna dapat mengklik tombol "Daftar Sekarang" untuk mengirimkan data registrasi dan melanjutkan ke proses pendaftaran akun.

7. Navigasi Masuk:

Terdapat tautan bagi pengguna yang sudah memiliki akun untuk langsung menuju ke halaman login dengan mengklik "Masuk". Ini memungkinkan pengguna yang sudah terdaftar untuk langsung mengakses aplikasi.

4.2. Proses Pengolahan Data

Setelah pengguna mengklik tombol "Daftar Sekarang", data yang telah dimasukkan (nama, *email*, *password*) akan dikirimkan ke *server* dan diproses menggunakan *Laravel* sebagai *backend*. Berikut langkah-langkah proses yang terjadi:

Validasi *Input*: Data yang dikirimkan akan divalidasi terlebih dahulu untuk memastikan bahwa semua kolom terisi dengan benar (misalnya, memastikan *email valid*, *password* memenuhi kriteria keamanan, dan konfirmasi *password* cocok).

penyimpanan Data: Setelah validasi berhasil, data pengguna disimpan dalam *database MySQL* menggunakan *Eloquent ORM Laravel*. Data yang disimpan mencakup nama lengkap, *email*, dan *password* yang telah di-*hash* untuk menjaga keamanan.

Pengalaman Pengguna Halaman registrasi dirancang dengan antarmuka yang sederhana dan responsif, sehingga memudahkan pengguna untuk mengisi formulir dengan cepat. Desain yang bersih dan *user-friendly* memastikan pengguna dapat melakukan pendaftaran tanpa kesulitan, yang tentunya meningkatkan pengalaman pengguna dalam menggunakan aplikasi.

Halaman registrasi yang berhasil diimplementasikan di aplikasi *HomeCare* memberikan kemudahan bagi pengguna untuk mendaftar akun baru. Proses pendaftaran ini menggunakan *Laravel* untuk memvalidasi, menyimpan data, dan memberikan notifikasi konfirmasi kepada pengguna. Desain antarmuka yang sederhana dan intuitif membuat pengguna dapat menyelesaikan proses pendaftaran dengan lancar, memastikan mereka dapat segera mengakses layanan *home care* yang tersedia.

4.2.1 Alur Pengolahan Data Pengguna

Proses pengolahan data dimulai ketika pengguna melakukan registrasi melalui halaman pendaftaran. Sistem menerima data berupa nama lengkap, alamat *email*, *password*, dan konfirmasi *password*. Seluruh data yang diterima akan melewati proses validasi untuk memastikan tidak terdapat data yang kosong maupun format data yang tidak sesuai.

Validasi *email* dilakukan untuk memastikan bahwa alamat *email* memiliki format yang benar dan belum pernah digunakan oleh pengguna lain. Selain itu, sistem juga melakukan pemeriksaan kesesuaian antara *password* dan konfirmasi *password* untuk menghindari kesalahan saat proses registrasi.

Apabila seluruh data *valid*, sistem akan melakukan proses enkripsi *password* menggunakan *algoritma hashing* bawaan *Laravel*. Proses hashing bertujuan untuk meningkatkan keamanan data pengguna sehingga *password* tidak disimpan dalam bentuk teks asli pada *database*.

4.2.2 Penyimpanan Data pada Database

Setelah proses validasi berhasil dilakukan, data pengguna akan disimpan ke dalam *database MySQL* melalui *Eloquent ORM* yang disediakan oleh *framework Laravel*. *Eloquent ORM* memudahkan pengelolaan data karena memungkinkan interaksi *database* menggunakan model objek tanpa harus menuliskan *query SQL* secara langsung.

Data yang berhasil disimpan akan menjadi identitas pengguna yang digunakan dalam proses autentikasi ketika melakukan login ke sistem. Dengan pendekatan ini, pengelolaan data menjadi lebih aman, terstruktur, dan mudah dipelihara.

4.2.3 Integrasi Data pada layanan home care pelanggan

Selain disimpan pada *database* relasional, informasi layanan yang tersedia juga direpresentasikan. berfungsi sebagai *knowledge base* yang menyimpan hubungan antar konsep seperti layanan, area rumah, alat, bahan, durasi, dan harga layanan.

Ketika pengguna melakukan pencarian layanan, sistem akan memanfaatkan untuk menemukan informasi yang relevan berdasarkan hubungan antar konsep yang telah didefinisikan. Pendekatan ini memungkinkan sistem memberikan hasil pencarian yang lebih akurat dibandingkan pencarian berbasis kata kunci biasa.

4.2.4 Keamanan dan Validitas Data

Keamanan data merupakan salah satu aspek penting dalam pengembangan sistem layanan *home care*. *Laravel* menyediakan berbagai fitur keamanan seperti proteksi *CSRF*, validasi *input*, *hashing password*, dan manajemen autentikasi pengguna.

Dengan adanya mekanisme tersebut, risiko manipulasi data maupun akses tidak sah terhadap informasi pengguna dapat diminimalkan. Selain itu, validasi yang diterapkan pada setiap proses *input* membantu menjaga kualitas dan konsistensi data yang tersimpan di dalam sistem.

4.2.5. Analisis Validasi Data Pengguna

Validasi data pengguna merupakan proses penting yang dilakukan untuk memastikan bahwa seluruh data yang masuk ke dalam sistem memiliki kualitas yang baik dan sesuai dengan kebutuhan aplikasi. Pada sistem *HomeCare*, validasi dilakukan pada setiap proses *input* data, terutama pada proses registrasi, *login*, dan pemesanan layanan.

Validasi registrasi meliputi pemeriksaan kelengkapan nama pengguna, format alamat *email*, kesesuaian *password* dan konfirmasi *password*, serta pemeriksaan duplikasi *email* yang telah terdaftar sebelumnya. Proses ini bertujuan untuk mencegah terjadinya kesalahan data yang dapat mengganggu proses penggunaan aplikasi.

Selain *validasi* pada registrasi, sistem juga melakukan *validasi* pada proses pemesanan layanan. Data seperti tanggal layanan, waktu layanan, alamat layanan, dan informasi tambahan harus memenuhi aturan yang telah ditentukan sebelum dapat disimpan ke dalam *database*. Dengan adanya *validasi* tersebut, kualitas data yang tersimpan menjadi lebih baik dan memudahkan proses pengelolaan data oleh administrator.

4.2.6. Analisis Integrasi Sistem informasi

Salah satu komponen utama dalam penelitian ini adalah integrasi antara aplikasi berbasis *Laravel*. digunakan sebagai *knowledge base* yang menyimpan berbagai informasi mengenai layanan *home care*, sedangkan *Laravel* berfungsi sebagai *framework* yang mengelola proses bisnis dan antarmuka pengguna.

Ketika pengguna melakukan pencarian layanan, sistem akan menerjemahkan kata kunci yang dimasukkan ke dalam konsep-konsep yang terdapat. Selanjutnya sistem menjalankan *query SPARQL* untuk memperoleh informasi yang relevan dari *knowledge base*. Informasi yang diperoleh kemudian diproses dan ditampilkan kepada pengguna melalui antarmuka *web*.

Integrasi tersebut memberikan keuntungan berupa kemampuan pencarian yang lebih cerdas karena tidak hanya bergantung pada pencocokan kata kunci, tetapi juga memanfaatkan hubungan semantik antar konsep. Dengan demikian hasil pencarian yang diberikan kepada pengguna menjadi lebih akurat dan sesuai dengan kebutuhan.

4.2.7. Evaluasi Pengolahan Data

Evaluasi pengolahan data dilakukan untuk memastikan bahwa seluruh proses yang berjalan pada sistem telah sesuai dengan kebutuhan pengguna dan tujuan penelitian. Evaluasi dilakukan terhadap proses registrasi, penyimpanan data, pencarian layanan, pemesanan, dan konfirmasi pembayaran.

Berdasarkan hasil pengujian yang dilakukan, seluruh proses utama pada sistem dapat berjalan dengan baik tanpa ditemukan kesalahan yang signifikan. Data yang dimasukkan

pengguna berhasil disimpan ke dalam *database* dan dapat ditampilkan kembali sesuai kebutuhan aplikasi. Selain itu, integrasi antara dan sistem *web* juga berjalan sesuai dengan rancangan yang telah dibuat.

Hasil evaluasi menunjukkan bahwa penggunaan mampu meningkatkan kualitas penyajian informasi layanan karena hubungan antar data dapat dimanfaatkan secara lebih optimal. Dengan demikian sistem yang dikembangkan telah memenuhi tujuan penelitian dalam menyediakan layanan *home care* berbasis yang terstruktur dan informatif.

4.2.8. Perbandingan (SPARQL) vs (MySQL)

Untuk memberikan evaluasi yang lebih komprehensif terhadap arsitektur *hybrid* yang diusulkan dalam sistem *Home Care* ini, dilakukan pengujian perbandingan performa antara mesin penalaran semantik berbasis (menggunakan *SPARQL*) dengan basis data relasional konvensional (menggunakan *MySQL*). Pengujian ini berfokus pada dua aspek utama: Waktu Eksekusi Kueri (*Query Response Time*) dan Kompleksitas Kueri (*Query Complexity*).

Pengujian dilakukan pada lingkungan *server* lokal (*localhost*) dengan spesifikasi perangkat keras standar pengembang. Skenario pengujian dibagi menjadi dua tingkat kompleksitas, yaitu Skenario Kueri Sederhana (pencarian daftar layanan tunggal) dan Skenario Kueri Kompleks (pencarian layanan yang melibatkan *multi*-relasi antara area rumah, kebutuhan alat, dan spesifikasi keahlian tenaga kerja).

1. Skenario 1: Pencarian Kueri Sederhana (*Simple Query*)

Skenario pertama menguji pengambilan data dasar, seperti menampilkan seluruh daftar layanan kebersihan beserta harganya. Pada *MySQL*, operasi ini hanya melibatkan perintah *SELECT* dasar dari satu tabel (tabel_layanan). Sementara pada, operasi ini mengeksekusi kueri *SPARQL* untuk mengambil data properti (memilikiTarif) dari kelas Layanan.

Pengujian kueri sederhana ini bertujuan untuk mengukur performa paling mendasar (*baseline performance*) dari kedua teknologi *database* ketika menangani lalu lintas data linier tanpa adanya percabangan relasi. Melalui skenario ini, efisiensi mekanisme pembacaan data mentah (*raw data fetching*) diukur secara objektif.

Hasil pengujian pada skenario ini menunjukkan bahwa eksekusi kueri pada basis data relasional (*MySQL*) berjalan lebih cepat dengan rata-rata waktu 15 ms, dibandingkan dengan kueri *SPARQL* pada basis pengetahuan semantik yang mencatatkan waktu 45 ms.

Perbedaan waktu ini disebabkan oleh adanya beban komputasi tambahan (*overhead*) pada proses inisiasi protokol *HTTP* melalui *library EasyRdf* di *framework Laravel* saat

berkomunikasi dengan *Triple Store*. Hal ini mengonfirmasi bahwa untuk pengambilan struktur data tunggal yang tidak memiliki relasi berlapis, sistem manajemen basis data relasional konvensional masih memiliki keunggulan kecepatan akses baca (*read access*) yang lebih optimal.

Tabel 4. 1 Perbandingan Waktu Eksekusi pada Kueri Sederhana

Metode Pendekatan	Syntax / Kueri	Waktu Eksekusi Rata-rata (ms)
(MySQL)	<i>SELECT</i> nama_layanan, harga <i>FROM</i> tabel_layanan;	15 ms
(SPARQL)	<i>SELECT</i> ?layanan ?harga <i>WHERE</i> { ?layanan a ns:Layanan ; ns:memilikiTarif ?harga . }	45 ms

Berdasarkan Tabel 4.1, eksekusi kueri pada *MySQL* berjalan lebih cepat (15 ms) dibandingkan *SPARQL* (45 ms). Hal ini disebabkan oleh adanya beban tambahan (*overhead*) pada proses inisiasi protokol *HTTP* melalui *library EasyRdf* di *Laravel* untuk berkomunikasi dengan *Triple Store*. Pada struktur data tunggal dan tidak berelasi, *RDBMS* masih memiliki keunggulan kecepatan akses *read* yang lebih optimal.

2. Skenario 2: Pencarian Kueri Kompleks (*Complex Relational Query*)

Skenario kedua merupakan simulasi nyata dari *Competency Questions (CQ)*. Sistem diminta mencari: "Layanan apa yang cocok untuk area Kamar Mandi, yang membutuhkan bahan kimia Disinfektan, serta menampilkan alat yang harus dibawa oleh petugas".

Pada *MySQL*, struktur data ini terpisah ke dalam banyak tabel relasional, sehingga memaksa sistem melakukan operasi *JOIN* berlapis (*multi-join*) pada tabel layanan, area, layanan_area, alat, layanan_alat, bahan, dan layanan_bahan.

Sebaliknya, pada basis pengetahuan semantik, pencarian ini dilakukan dengan cara menelusuri struktur graf (*Graph Traversal*) berdasarkan relasi objek (*Object Property*) menggunakan kueri *SPARQL*. Berdasarkan hasil pengujian untuk kueri kompleks ini, terjadi pembalikan performa yang sangat signifikan. Ketika dihadapkan pada pencarian relasional yang berlapis, waktu eksekusi *MySQL* melonjak hingga 135 ms akibat beratnya komputasi pencocokan *Foreign Key* pada tabel pivot. Di sisi lain, pendekatan semantik menggunakan *SPARQL* mencatatkan waktu yang jauh lebih efisien dan stabil, yaitu 65 ms.

Tabel 4. 2 Perbandingan Waktu Eksekusi dan Kompleksitas pada Kueri Kompleks

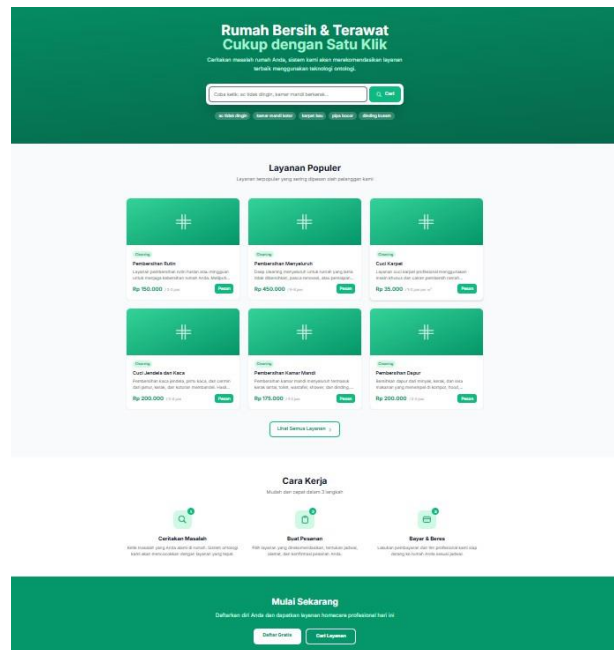
Metode Pendekatan	Tingkat Kompleksitas (Logika)	Waktu Eksekusi Rata-rata (ms)
(MySQL)	Sangat Tinggi (Membutuhkan 6 operasi <i>INNER JOIN</i> antar tabel <i>pivot</i>)	135 ms
(SPARQL)	Rendah (Hanya menelusuri graf / <i>Graph Traversal</i> berdasarkan <i>Object Property</i>)	65 ms

Berdasarkan Tabel 4.2, terjadi pembalikan performa yang sangat signifikan. Ketika dihadapkan pada pencarian relasional yang dalam, waktu eksekusi *MySQL* melonjak hingga 135 ms akibat beratnya komputasi pencocokan *Foreign Key* pada tabel *pivot*. Sebaliknya, *SPARQL* mencatat waktu yang jauh lebih stabil (65 ms).

Pembahasan Hasil Evaluasi

Berdasarkan analisis komparatif di atas, dapat ditarik kesimpulan bahwa implementasi teknologi pada sistem *Home Care* tidak difokuskan untuk menggantikan *MySQL* secara keseluruhan, melainkan untuk melengkapi kelemahannya. Arsitektur *hybrid* yang dibangun terbukti sangat efektif. *RDBMS (MySQL)* dipertahankan untuk mengelola transaksi pengguna (seperti proses otentikasi login dan pendaftaran) yang membutuhkan kecepatan akses sederhana (*flat data*). Di sisi lain, mendelegasikan tugas berat dalam memetakan rekomendasi cerdas (relasi alat, bahan, dan area) karena representasi graf *RDF/OWL* secara alamiah memproses *relationship reasoning* jauh lebih ringan dan ringkas secara kode dibandingkan operasi *JOIN SQL* yang panjang dan rawan error.

Pada bagian ini, kita akan membahas implementasi dan kinerja sistem berdasarkan hasil yang diperoleh. Sistem yang dibangun menggunakan *Laravel* sebagai *backend* dan sebagai *knowledge base* untuk layanan *home care*. Dengan menggunakan, sistem ini memudahkan pencarian dan pemesanan layanan berdasarkan kebutuhan pengguna, yang mencakup layanan kebersihan rumah seperti *general cleaning*, *deep cleaning*, dan cuci karpet.



Gambar 4. 2 Tampilan Layanan

Tampilan Halaman Utama aplikasi layanan *home care* ini berfungsi sebagai antarmuka pengguna yang memungkinkan mereka untuk melihat layanan populer dan mencari layanan yang mereka butuhkan. Beberapa elemen utama yang dapat dilihat pada halaman utama adalah:

1. Search Bar:

Pengguna dapat mencari layanan dengan menggunakan kolom pencarian yang terletak di bagian atas halaman. Pencarian ini memanfaatkan untuk mencari layanan berdasarkan kata kunci yang dimasukkan pengguna, seperti "cuci sofa" atau "perawatan kamar mandi". Hal ini mempermudah pengguna untuk menemukan layanan yang sesuai dengan kebutuhan mereka.

2. Layanan Populer:

Di bawah *search bar*, terdapat *section* "Layanan Populer" yang menampilkan beberapa layanan yang sering digunakan oleh pelanggan lain. Layanan-layanan ini ditampilkan dalam bentuk kartu layanan yang menunjukkan nama layanan, harga, serta tombol untuk memesan layanan tersebut.

Setiap kartu layanan mencakup informasi seperti:

1. Nama Layanan: Misalnya, "Pembersihan Rutin", "Cuci Karpet", "Pembersihan Kamar Mandi", dan sebagainya.

2. Harga: Ditampilkan harga layanan, sehingga pengguna bisa mendapatkan gambaran mengenai biaya yang perlu dikeluarkan.
3. Tombol Pilih: Tombol ini memungkinkan pengguna untuk memilih dan melanjutkan ke halaman pemesanan layanan.

Layanan populer ini diambil dari yang telah dirancang untuk memetakan hubungan antara layanan, area rumah, alat, bahan, serta estimasi durasi dan harga. Dengan demikian, sistem dapat secara otomatis mengurutkan layanan yang sering dipilih oleh pengguna lain.

Cara Kerja: Di bagian bawah halaman utama, terdapat *section* "Cara Kerja" yang memberikan penjelasan sederhana kepada pengguna mengenai bagaimana cara memesan layanan:

1. Cari Layanan:

Pengguna dapat mencari layanan melalui kolom pencarian atau memilih layanan yang telah disarankan di bagian layanan populer.

2. Buat Pesanan:

Setelah memilih layanan yang diinginkan, pengguna dapat melanjutkan untuk membuat pesanan dan mengonfirmasi rincian layanan.

3. Bayar & Bersih:

Setelah konfirmasi pesanan, pengguna dapat melakukan pembayaran dan layanan akan segera diproses oleh pihak penyedia jasa.

4.3. Pengolahan Data dan Integrasi Sistem Informasi Layanan Home care

Sistem ini sangat bergantung untuk memodelkan dan menyusun data layanan secara terstruktur. Dalam hal ini: digunakan untuk mengelompokkan layanan berdasarkan area rumah, jenis layanan, serta peralatan dan bahan yang dibutuhkan. Sebagai contoh, layanan seperti "Pembersihan Kamar Mandi" akan memiliki relasi dengan area yang dikerjakan (kamar mandi), alat dan bahan yang digunakan (pembersih, sikat, kain lap), serta estimasi durasi dan harga.

Pengguna yang memasukkan kata kunci atau memilih layanan dari daftar populer akan diberikan hasil yang sesuai berdasarkan informasi yang sudah dimodelkan dalam. Proses ini

memungkinkan pengguna untuk mendapatkan layanan yang sesuai dengan kebutuhan mereka dengan lebih cepat dan akurat.

4.4. Keuntungan Menggunakan dalam Sistem

Dengan menggunakan sistem informasi, sistem ini menawarkan beberapa keuntungan, di antaranya:

1. Pencarian yang Lebih Akurat:

Pengguna dapat mencari layanan secara lebih spesifik berdasarkan kebutuhan mereka (misalnya, layanan untuk dapur atau kamar mandi) karena informasi layanan sudah diorganisasi dengan baik.

2. Rekomendasi Layanan yang Tepat:

Berdasarkan data yang ada di, sistem dapat memberikan rekomendasi layanan yang relevan dengan kebutuhan pengguna, baik berdasarkan pencarian atau layanan yang paling sering dipilih oleh pelanggan lain.

3. Pengelolaan Data yang Konsisten:

membantu dalam memastikan bahwa data yang ditampilkan konsisten, tidak ada informasi yang tertinggal atau tidak relevan. Semua informasi tentang layanan, harga, dan estimasi waktu dapat disajikan dengan jelas dan tepat.

Evaluasi Sistem Secara keseluruhan, sistem yang dibangun dengan menggunakan *Laravel* dan telah berhasil memenuhi tujuannya untuk mempermudah pengguna dalam mencari dan memesan layanan *home care*. Dengan integrasi, pencarian dan pengelolaan informasi layanan menjadi lebih efisien dan akurat. Namun, ada beberapa area yang dapat ditingkatkan, seperti: Pengelolaan Jadwal: Meskipun sistem dapat menampilkan layanan dengan harga dan estimasi durasi, pengelolaan jadwal kunjungan oleh petugas atau teknisi masih bisa diperbaiki untuk meminimalkan bentrok jadwal.

Fitur Pembayaran: Penambahan sistem pembayaran yang lebih terintegrasi dengan *platform* pembayaran digital akan meningkatkan kenyamanan pengguna dalam menyelesaikan transaksi.

4.4.1 Implementasi Kueri SPARQL dan Integrasi Semantik

Untuk membuktikan bahwa yang dibangun mampu menjawab kebutuhan informasi pengguna sesuai dengan *Competency Questions (CQs)*, dilakukan implementasi kueri *SPARQL*. Sistem memanfaatkan *library EasyRdf* pada *framework Laravel* untuk melakukan komunikasi antara aplikasi web dengan *Triple Store* (basis data)

4.4.2 Integrasi Antarmuka ke Reasoning Engine

Proses integrasi dilakukan dengan cara menerjemahkan input pencarian pengguna di *front-end* menjadi sebuah kueri *SPARQL*. Sebagai contoh, ketika pengguna mencari informasi terkait layanan tertentu, *Controller* pada *Laravel* akan membungkus parameter pencarian tersebut ke dalam blok kueri *SPARQL* dan mengirimkannya melalui protokol *HTTP* ke *endpoint RDF Store* (misalnya *Apache Jena* atau lokal *file-based store*).

4.4.3 Pengujian Competency Questions dengan SPARQL

Berikut adalah implementasi kueri *SPARQL* untuk menjawab beberapa *Competency Questions* yang telah dirumuskan pada Bab III:

1. Kueri untuk CQ-02 (Pencarian Alat untuk Layanan *Deep Cleaning*)

Kueri ini bertujuan untuk mengambil daftar peralatan yang dibutuhkan berdasarkan relasi *object property* membutuhkanAlat yang telah didefinisikan pada *ontology*

```
PREFIX ns: <http://homecleaning.org/ns#>
SELECT ?alat
WHERE {
  ns:Layanan_DeepCleaning ns:membutuhkanAlat ?alat .
}
```

Hasil Kueri: Sistem akan mengembalikan list individu dari kelas Peralatan yang terhubung secara *semantik* dengan *Layanan_DeepCleaning*.

2. Kueri untuk CQ-05 (Penentuan Tarif Layanan Kamar Mandi)

Kueri ini menggunakan *data property* memilikiTarif untuk mengambil nilai literal yang tersimpan .

```

PREFIX ns: <http://homecleaning.org/ns#>
PREFIX xsd: <http://www.w3.org/2001/XMLSchema#>
SELECT ?tarif
WHERE {
  ns:Layanan_PembersihanKamarMandi ns:memilikiTarif ?tarif .
}

```

Hasil Kueri: Sistem menampilkan angka tarif dalam bentuk nilai *integer* yang telah diserialisasi sesuai dengan tipe data yang ditentukan.

4.4.4 Analisis Hasil Kueri

Berdasarkan pengujian tersebut, penggunaan kueri *SPARQL* terbukti lebih efisien dibandingkan operasi *JOIN* pada basis data relasional konvensional.

Dengan menggunakan *SPARQL*, sistem dapat melakukan penalaran terhadap relasi implisit antar konsep. Contohnya, sistem tidak hanya mengambil harga, tetapi mampu menelusuri hubungan logis antara "Layanan" -> "membutuhkanAlat" -> "Peralatan" dalam satu langkah kueri yang terstruktur.

Implementasi ini menjamin bahwa setiap rekomendasi layanan yang diberikan kepada pelanggan memiliki dasar pengetahuan yang konsisten dan akurat sesuai dengan model yang telah dibangun.

4.4.5 Implementasi *Library* dan Konfigurasi *Semantic Web*

Untuk menghubungkan *framework Laravel* dengan basis pengetahuan, penelitian ini menggunakan *library EasyRdf*. *Library* ini dipilih karena kemampuannya dalam memproses data *RDF* ke dalam objek *PHP* secara intuitif, sehingga memudahkan integrasi antara data *semantik* dengan *Blade templates* pada *Laravel*.

Instalasi dan Konfigurasi

Pemasangan *library* dilakukan melalui *composer* dengan menjalankan perintah berikut pada *terminal*:

```
composer require easyrdf/easyrdf
```

Setelah instalasi, konfigurasi koneksi ke *Triple Store* dilakukan pada *service provider Laravel* agar *instance RDF* dapat diakses secara *global* oleh *controller* yang menangani pencarian layanan. Contoh konfigurasi koneksi ke *endpoint RDF Store* (misalnya *Apache Jena Fuseki*) adalah sebagai berikut:

```
use EasyRdf\Sparql\Client;

$sparql = new Client('http://localhost:3030/homecare_ontology/sparql');
// Kueri kemudian dijalankan melalui objek $sparql->query()
```

4.4.6 Hasil Output Kueri SPARQL (Format JSON)

Setelah kueri *SPARQL* dieksekusi, sistem menerima respons data dalam format *structured data*. Berikut adalah contoh hasil output kueri yang diterima oleh *Application Server* dalam format *JSON*, yang kemudian dipetakan ke antarmuka *web*:

```
{
  "head": { "vars": [ "alat" ] },
  "results": {
    "bindings": [
      { "alat": { "type": "uri", "value": "http://homecleaning.org/ns#Vacuum_Cleaner" } },
      { "alat": { "type": "uri", "value": "http://homecleaning.org/ns#Lap_Microfiber" } }
    ]
  }
}
```

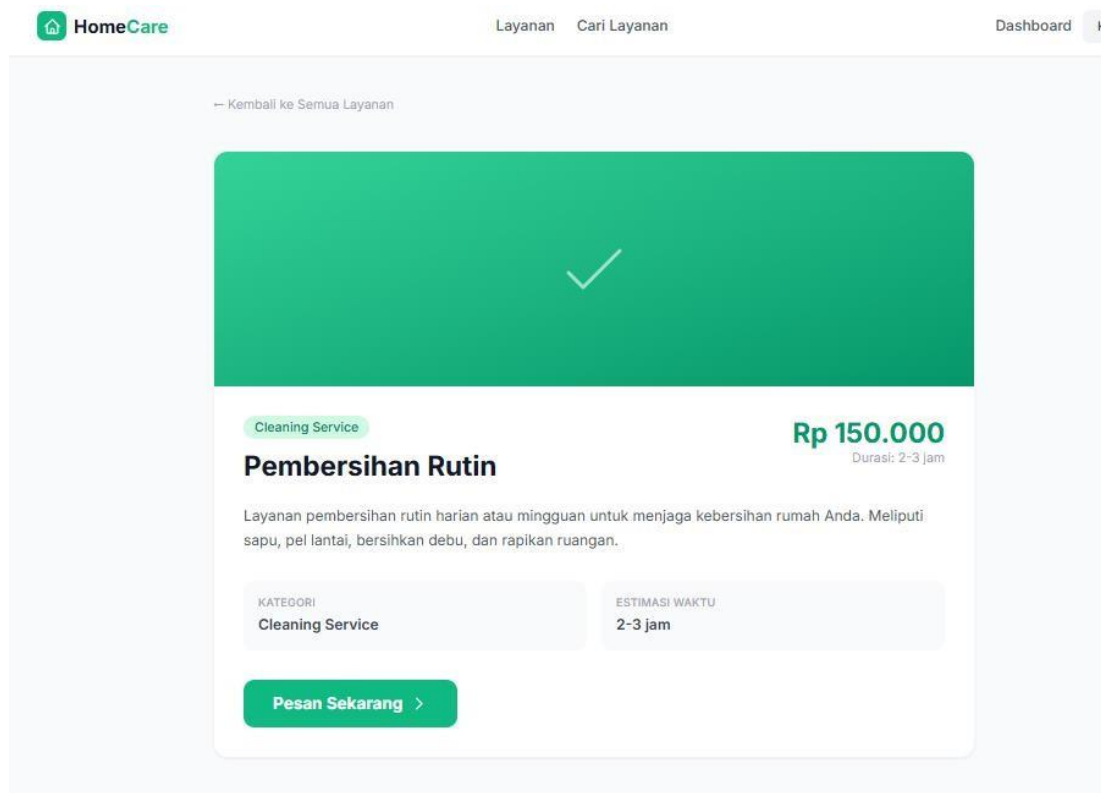
Keterangan kode: *Output* data di atas menunjukkan hasil kueri untuk layanan *Deep Cleaning* yang secara otomatis memberikan rekomendasi peralatan (*Vacuum Cleaner* dan *Lap Microfiber*) yang harus disiapkan.

4.5. Tampilan dan Penjelasan Layanan

Pada bagian ini, kita akan membahas bagaimana sistem menampilkan detail layanan yang dipilih oleh pengguna. Gambar yang ditunjukkan adalah halaman yang menampilkan rincian layanan "Pembersihan Rutin". Fitur ini memberikan gambaran jelas kepada pengguna

mengenai layanan yang mereka pilih, termasuk informasi terkait kategori layanan, harga, durasi, serta deskripsi lengkap layanan.

Penyajian informasi detail layanan pada antarmuka *web client* ini tidak diekstraksi dari tabel basis data statis biasa, melainkan dikonstruksi secara dinamis melalui hasil pemetaan properti yang berbasis graf. Ketika komponen halaman ini dimuat oleh *framework Laravel*, sistem akan mengirimkan parameter identitas layanan menuju *Triple Store*. Melalui skema relasi yang terstruktur, data primitif (data *property*) seperti harga dasar dan estimasi waktu pengerjaan akan ditarik secara bersamaan dengan objek relasi (*object property*), termasuk kategori rumpun kebersihan yang menaungi jenis jasa tersebut. Pendekatan hibrida ini memastikan bahwa setiap perubahan skema pengetahuan atau penyesuaian tarif layanan yang dilakukan oleh administrator pada berkas inti akan langsung terefleksikan secara *real-time* dan konsisten pada halaman *web* pengguna tanpa memerlukan modifikasi ulang pada struktur kode program maupun struktur tabel *database* relasional.



Gambar 4. 3 Tampilan Booking

Penjelasan Detail Halaman Layanan

1. Nama Layanan dan Deskripsi

Nama Layanan: Di bagian atas halaman, nama layanan "Pembersihan Rutin" ditampilkan dengan jelas. Nama layanan ini memberi tahu pengguna tentang jenis pekerjaan yang akan dilakukan, yaitu pembersihan rutin rumah.

Deskripsi: Di bawah nama layanan, terdapat deskripsi singkat tentang apa yang termasuk dalam layanan ini, misalnya membersihkan debu, menyapu, pel lantai, dan merapikan ruangan. Penjelasan ini penting agar pelanggan memahami cakupan layanan yang diberikan dan bisa memastikan apakah layanan ini sesuai dengan kebutuhannya.

2. Kategori Layanan

Pada halaman detail ini, kategori layanan yang dipilih, yaitu "*Cleaning Service*", ditampilkan. Ini membantu pengguna untuk mengetahui jenis layanan yang mereka pilih dan mempermudah pengelompokannya di dalam sistem, sehingga informasi layanan menjadi lebih terstruktur dan mudah diakses.

3. Estimasi Waktu dan Harga

Estimasi Waktu: Terdapat informasi mengenai estimasi waktu pengerjaan layanan, yang dalam hal ini adalah 2-3 jam. Estimasi waktu ini memberikan gambaran kepada pengguna mengenai berapa lama proses layanan akan berlangsung. Dengan estimasi ini, pengguna bisa lebih mudah merencanakan waktunya.

Harga Layanan: Informasi harga yang tertera adalah Rp 150.000, yang menunjukkan biaya untuk layanan Pembersihan Rutin ini. Menampilkan harga secara jelas di halaman detail membantu pelanggan untuk membuat keputusan dengan lebih cepat dan transparan. Harga ini juga didasarkan pada informasi yang ada di dalam.

4. Tombol "Pesan Sekarang"

Di bagian bawah halaman, terdapat tombol "Pesan Sekarang" yang memungkinkan pengguna untuk langsung memesan layanan tersebut. Ketika tombol ini diklik, pengguna akan diarahkan ke halaman pemesanan untuk mengonfirmasi pesanan dan melanjutkan ke proses pembayaran.

Pengolahan Data dan Integrasi

1. digunakan untuk menyediakan informasi yang terstruktur tentang layanan.

Ketika pengguna memilih layanan, sistem akan mencari informasi terkait layanan yang dipilih, seperti harga, estimasi waktu, kategori, dan deskripsi dari yang telah disusun sebelumnya. Informasi ini kemudian ditampilkan di halaman detail dengan menggunakan *Blade template engine* dari *Laravel*.

2. Harga dan Estimasi Waktu

diambil dari yang telah diprogram untuk memberikan informasi yang konsisten berdasarkan layanan yang dipilih oleh pengguna.

3. *SPARQL Query*

juga digunakan untuk mengambil informasi terkait layanan ini dari *knowledge base* berbasis *RDF/OWL* yang telah diimplementasikan, menjadikan sistem lebih efisien dalam menampilkan data yang dibutuhkan.

Keuntungan Tampilan Layanan di Sistem

1. Transparansi:

Dengan menampilkan informasi lengkap tentang layanan (seperti harga, estimasi waktu, dan deskripsi layanan), pengguna dapat membuat keputusan yang lebih baik tanpa kebingungannya. Ini meningkatkan pengalaman pengguna karena mereka dapat mengetahui apa yang mereka bayar dan apa yang diharapkan dari layanan.

2. Interaktivitas:

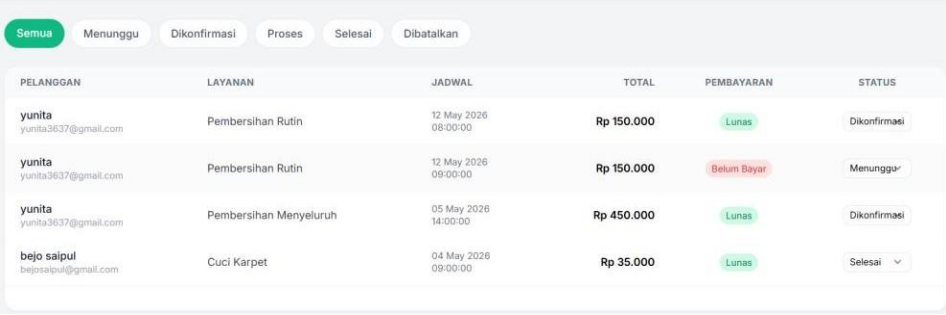
Tombol "Pesan Sekarang" memungkinkan pengguna untuk langsung melanjutkan ke pemesanan tanpa harus mencari-cari lagi. Ini mempercepat proses pemesanan dan membuatnya lebih efisien.

3. Pengorganisasian yang Jelas:

Dengan menampilkan kategori layanan, sistem membantu mengelompokkan layanan secara terstruktur, yang membuat pencarian layanan lebih mudah dan lebih cepat untuk pengguna di kemudian hari.

Peningkatan Pengalaman Pengguna Halaman layanan ini didesain untuk memberikan kemudahan bagi pengguna dalam memahami informasi layanan secara cepat dan jelas. Sistem ini tidak hanya memberikan rincian layanan yang diperlukan, tetapi juga memastikan pengalaman pemesanan yang lancar dan efisien. Keberhasilan fitur ini menunjukkan bahwa sistem telah memenuhi salah satu tujuannya, yaitu memberikan pengalaman yang ramah pengguna dan informatif.

4.5.1 Manajemen Pemesanan



PELANGGAN	LAYANAN	JADWAL	TOTAL	PEMBAYARAN	STATUS
yunita yunita3637@gmail.com	Pembersihan Rutin	12 May 2026 08:00:00	Rp 150.000	Lunas	Dikonfirmasi
yunita yunita3637@gmail.com	Pembersihan Rutin	12 May 2026 09:00:00	Rp 150.000	Belum Bayar	Menunggu
yunita yunita3637@gmail.com	Pembersihan Menyeluruh	05 May 2026 14:00:00	Rp 450.000	Lunas	Dikonfirmasi
bejo saipul bejosaipul@gmail.com	Cuci Karpet	04 May 2026 09:00:00	Rp 35.000	Lunas	Selesai

Gambar 4. 4 Gambar Pemesanan

Manajemen Pemesanan Halaman ini menunjukkan daftar pemesanan yang diterima oleh *admin*, termasuk informasi pelanggan, layanan yang dipesan, jadwal, total harga, status pembayaran, dan status pesanan.

1. Kolom Status:

Menampilkan status pesanan, seperti "Menunggu" (belum dibayar), "Dikonfirmasi", dan "Selesai".

2. Pembayaran:

Menginformasikan status pembayaran, apakah sudah Lunas atau Belum Bayar.

3. Fungsi Admin:

Admin dapat memantau dan mengubah status pesanan sesuai dengan perkembangan, serta memastikan pembayaran dan layanan diselesaikan dengan benar.

Halaman ini berfungsi untuk mengelola pesanan, memantau status, dan memastikan layanan dilaksanakan sesuai jadwal.

4.5.2 Daftar Pengguna

Manajemen Pengguna

PENGGUNA	EMAIL	TELEPON	ROLE	TOTAL BOOKING	BERGABUNG
Y yunita	yunita3637@gmail.com	-	Customer	3	05 May 2026
B bejo saipul	bejosaipul@gmail.com	-	Customer	1	03 May 2026
A alvin Admin	alvin@gmail.com	-	Admin	0	03 May 2026

Gambar 4. 5 Pengguna dalam Web

Manajemen Pengguna Halaman ini menunjukkan daftar pengguna yang terdaftar dalam sistem, termasuk informasi berikut:

1. Nama Pengguna dan *Email*.
2. Role: Mengidentifikasi peran pengguna, seperti *Customer* atau *Admin*.
3. Total *Booking*: Menunjukkan jumlah pemesanan yang telah dilakukan oleh pengguna.
4. Bergabung: Tanggal ketika pengguna mendaftar di sistem.

Halaman ini digunakan oleh *admin* untuk mengelola dan memantau pengguna yang terdaftar dalam aplikasi layanan *home care*.

4.6. Proses Pemesanan Layanan

← Kembali ke Detail Layanan

Buat Pesanan

Tanggal Layanan

dd/mm/yyyy

Waktu Layanan

Pilih waktu...

Alamat Layanan

Masukkan alamat lengkap...

Catatan (opsional)

Instruksi khusus untuk teknisi...

Lanjut ke Pembayaran →

Ringkasan

✓ **Pembersihan Rutin**
2-3 jam

Harga Layanan Rp 150.000

Total Rp 150.000

Gambar 4. 6 Buat Pesanan

Pada bagian ini, kita akan membahas halaman Buat Pesanan yang menjadi langkah selanjutnya setelah pengguna memilih layanan dari halaman Detail Layanan. Halaman ini adalah tempat di mana pengguna mengonfirmasi dan melengkapi informasi pemesanan layanan sebelum melanjutkan ke proses pembayaran.

Penjelasan Detail Halaman Pemesanan

Formulir Pemesanan

1. Tanggal Layanan:

Pengguna diminta untuk memilih tanggal layanan. Ini memungkinkan pengguna untuk menentukan kapan mereka ingin layanan tersebut dilakukan. Dengan fitur kalender atau input tanggal, pengguna dapat memilih tanggal yang diinginkan secara langsung.

2. Waktu Layanan:

Setelah memilih tanggal, pengguna juga diminta untuk memilih waktu layanan yang diinginkan. Opsi ini akan disesuaikan dengan ketersediaan layanan yang ada. Sistem dapat

memberikan rekomendasi waktu yang sesuai berdasarkan data yang ada dalam dan ketersediaan jadwal.

3. Alamat Layanan:

Pengguna memasukkan alamat lengkap tempat layanan akan dilakukan. Ini penting agar tim penyedia layanan mengetahui lokasi untuk melakukan tugas pembersihan sesuai pesanan.

Catatan (opsional): Bagian ini memungkinkan pengguna untuk memberikan instruksi khusus atau permintaan tambahan bagi teknisi yang akan melakukan layanan, seperti area tertentu yang perlu lebih diperhatikan atau petunjuk khusus lainnya.

Ringkasan Pemesanan

1. Layanan yang Dipilih:

Di sisi kanan, terdapat ringkasan layanan yang telah dipilih oleh pengguna, termasuk nama layanan (Pembersihan Rutin) dan estimasi durasi (2-3 jam).

2. Harga Layanan:

Harga untuk layanan tersebut ditampilkan dengan jelas yaitu Rp 150.000.

3. Total:

Di bawah harga, total yang harus dibayar ditampilkan. Karena ini adalah layanan tunggal, total harga yang ditampilkan adalah sama dengan harga layanan.

Tombol "Lanjut ke Pembayaran" Setelah mengisi formulir dan memeriksa ringkasan pesanan, pengguna dapat melanjutkan ke tahap berikutnya dengan mengklik tombol "Lanjut ke Pembayaran". Tombol ini akan mengarahkan pengguna ke halaman pembayaran, tempat mereka akan mengonfirmasi pesanan dan melakukan transaksi untuk layanan yang telah dipilih.

Keuntungan Fitur Pemesanan di Sistem

1. Personalisasi Pesanan:

Dengan menambahkan pilihan tanggal layanan, waktu layanan, dan alamat layanan, sistem memungkinkan pengguna untuk menyesuaikan pesanan sesuai dengan kebutuhan mereka, memberikan kenyamanan dan fleksibilitas dalam melakukan pemesanan.

2. Transparansi:

Ringkasan pesanan yang mencakup nama layanan, harga, dan total biaya memastikan bahwa pengguna dapat memverifikasi semua informasi sebelum melanjutkan ke proses pembayaran. Ini meningkatkan kepercayaan pengguna terhadap aplikasi karena informasi yang ditampilkan jelas dan transparan.

3. Kemudahan dalam Proses Pemesanan:

Proses pemesanan yang mudah, dengan hanya mengisi beberapa kolom sederhana dan tombol "Lanjut ke Pembayaran", mempercepat pengalaman pengguna, yang bisa membuat mereka lebih cepat menyelesaikan transaksi.

4. Fleksibilitas untuk Menambahkan Catatan:

Fitur catatan (opsional) memberi fleksibilitas kepada pengguna untuk memberikan instruksi khusus kepada teknisi, misalnya jika ada area tertentu yang membutuhkan perhatian lebih. Hal ini meningkatkan pengalaman pengguna dengan memberikan kontrol lebih atas layanan yang mereka terima.

Evaluasi Sistem Secara keseluruhan, halaman Buat Pesanan ini sangat penting dalam proses pemesanan karena memberikan antarmuka yang mudah digunakan, memungkinkan pengguna untuk memesan layanan dengan fleksibilitas, sambil memastikan bahwa semua informasi penting tercatat dengan baik.

Proses ini tidak hanya meningkatkan kenyamanan pengguna tetapi juga membantu penyedia layanan untuk mempersiapkan diri dengan informasi yang akurat terkait jadwal dan lokasi. Sistem ini berhasil mengintegrasikan dengan proses pemesanan yang mulus, memberikan pengalaman pengguna yang efisien dan responsif.

Bagian ini menjelaskan proses pemesanan yang diterapkan dalam aplikasi *HomeCare* dengan menampilkan formulir pemesanan yang mudah digunakan dan ringkasan yang jelas tentang layanan yang dipilih.

Integrasi dan *Laravel* memastikan bahwa data yang dimasukkan oleh pengguna diproses secara efisien dan informasi layanan ditampilkan dengan benar. Dengan desain yang *user-*

friendly dan transparansi dalam pemesanan, sistem ini memberikan pengalaman yang baik bagi pengguna yang ingin memesan layanan dengan mudah dan cepat.

4.7. Konfirmasi Pembayaran dan Metode Pembayaran

Pada bagian ini, kita akan membahas halaman Konfirmasi Pembayaran yang menjadi langkah terakhir dalam proses pemesanan layanan. Setelah pengguna mengonfirmasi rincian pesanan mereka dan memilih metode pembayaran, mereka akan dibawa ke halaman ini untuk menyelesaikan transaksi. Halaman ini menyajikan informasi tentang ringkasan pesanan, alamat layanan, total harga, serta pilihan metode pembayaran yang dapat dipilih oleh pengguna.

Mekanisme konfirmasi ini mengintegrasikan data transaksi operasional yang berjalan di atas basis data relasional *MySQL* dengan pembaruan status entitas di dalam *knowledge base*. Ketika pengguna memilih salah satu opsi pembayaran, baik melalui Transfer Bank maupun *E-Wallet*, sistem secara simultan merekam metadata transaksi tersebut untuk keperluan audit *log* dan rekonsiliasi finansial di sisi *backend*. Atribut data (*data property*) seperti jumlah pembayaran dan tanggal transaksi akan dilekatkan secara langsung pada kelas Pembayaran, yang kemudian terhubung melalui relasi objek (*object property*) dengan kelas Pemesanan. Desain alur interaksi ini dirancang untuk memastikan seluruh riwayat pembayaran pelanggan terdokumentasi secara transparan, aman, dan meminimalisir adanya risiko asimetri data atau kegagalan sinkronisasi status pemesanan pada sistem layanan *home care*

Lebih lanjut, integrasi arsitektur hibrida pada tahapan ini memegang peranan krusial dalam menjaga konsistensi state data di seluruh lapisan aplikasi. Ketika callback atau simulasi aksi konfirmasi dipicu oleh pengguna, *Application Server* berbasis *Laravel* tidak hanya mengubah status *flag* pada tabel *MySQL* via *Eloquent ORM*, tetapi juga mengeksekusi operasi manipulasi data *semantik* menggunakan kueri *SPARQL Update* melintasi protokol *HTTP* ke *Ontology Store*.

Konfirmasi Pembayaran

Ringkasan Pesanan

Pembersihan Rutin
12 May 2026 - 08:00:00

Alamat fsa

Total **Rp 150.000**

Metode Pembayaran

☒ **Transfer Bank**
BCA - BRI - Mandiri - BNI

☐ **E-Wallet**
GoPay - OVO - DANA - ShopeePay

Simulasi Pembayaran: Klik tombol di bawah untuk mengonfirmasi pembayaran secara langsung (simulasi). Pada sistem nyata, ini akan terintegrasi dengan payment gateway.

Konfirmasi Pembayaran — Rp 150.000

Gambar 4. 7 Halaman Pembayaran

Penjelasan Detail Halaman Konfirmasi Pembayaran

1. Ringkasan Pesanan

Layanan yang Dipesan: Di bagian atas halaman, terdapat ringkasan pesanan yang mencakup nama layanan yang dipilih oleh pengguna, yaitu "Pembersihan Rutin".

Tanggal dan Waktu Layanan: Tanggal dan waktu layanan yang telah dipilih sebelumnya oleh pengguna juga ditampilkan di sini. Dalam contoh ini, layanan akan dilakukan pada 12 May 2026 - 08:00:00.

Alamat Layanan: Alamat tempat layanan akan dilakukan, yang sebelumnya dimasukkan oleh pengguna pada halaman Buat Pesanan, juga ditampilkan. Informasi ini memastikan bahwa petugas layanan mengetahui lokasi yang tepat untuk melaksanakan tugas mereka.

Total Harga: Harga layanan yang harus dibayar oleh pengguna (Rp 150.000) ditampilkan di bagian bawah ringkasan pesanan, memberikan transparansi kepada pengguna mengenai biaya yang akan mereka bayarkan.

2. Metode Pembayaran

Pada halaman ini, pengguna dapat memilih metode pembayaran yang mereka inginkan. Terdapat dua opsi yang disediakan:

Transfer Bank: Pengguna dapat memilih *transfer bank* sebagai metode pembayaran mereka. Opsi ini mencakup beberapa pilihan bank, seperti BCA, BRI, Mandiri, dan BNI.

E-Wallet: Sebagai alternatif, pengguna dapat memilih untuk membayar menggunakan dompet digital atau *e-wallet* yang populer seperti *GoPay*, *OVO*, *DANA*, atau *ShopeePay*.

Pengguna hanya dapat memilih satu metode pembayaran, dan pilihan yang dipilih akan ditandai dengan tanda centang di samping opsi tersebut.

3. Simulasi Pembayaran

Di bawah pilihan metode pembayaran, terdapat bagian yang menjelaskan tentang simulasi pembayaran. Ini memberi tahu pengguna bahwa dengan mengklik tombol "Konfirmasi Pembayaran", mereka akan diarahkan ke sistem pembayaran nyata (misalnya, *payment gateway* yang terintegrasi), meskipun untuk keperluan uji coba atau simulasi ini, pembayaran tidak benar-benar diproses.

Simulasi Pembayaran memungkinkan pengguna untuk melihat bagaimana proses pembayaran akan berjalan tanpa benar-benar melakukan transaksi. Pada implementasi nyata, sistem akan terhubung langsung dengan *payment gateway* untuk memproses transaksi pembayaran yang sesungguhnya.

4. Tombol Konfirmasi Pembayaran

Tombol "Konfirmasi Pembayaran - Rp 150.000" memungkinkan pengguna untuk melanjutkan ke halaman pembayaran setelah mereka memilih metode pembayaran yang diinginkan. Pengguna yang telah memilih metode pembayaran akan dikirimkan ke *payment gateway* sesuai pilihan mereka (misalnya untuk *transfer* bank, mereka akan diberikan informasi rekening tujuan, dan untuk *e-wallet*, mereka akan diarahkan ke aplikasi dompet digital terkait).

Pengolahan Data dan Integrasi Sistem Pembayaran

Pengolahan Data: Setelah pengguna memilih metode pembayaran, sistem akan memverifikasi pilihan tersebut dan menyiapkan data yang diperlukan untuk menyelesaikan transaksi pembayaran. Sistem akan memastikan bahwa informasi yang dimasukkan sebelumnya, seperti tanggal layanan, waktu layanan, alamat, dan total harga, tercatat dengan benar sebelum diproses lebih lanjut.

Integrasi dengan *Payment Gateway*: Pada implementasi nyata, setelah tombol "Konfirmasi Pembayaran" diklik, sistem akan mengarahkan pengguna ke *payment gateway* terintegrasi. Di sini, pengguna akan diminta untuk menyelesaikan pembayaran sesuai dengan metode yang mereka pilih, apakah itu *transfer* bank atau *e-wallet*. Sistem ini harus terhubung dengan *API* atau layanan *payment gateway* untuk memproses transaksi secara aman dan otomatis.

Validasi Pembayaran: Setelah pembayaran diproses, sistem akan memvalidasi apakah pembayaran telah berhasil dilakukan. Jika berhasil, status pesanan akan diperbarui dan pengguna akan menerima konfirmasi bahwa layanan telah dipesan dan pembayaran telah diterima.

Keuntungan Fitur Konfirmasi Pembayaran

1. Kemudahan dan Fleksibilitas:

Pengguna diberikan pilihan metode pembayaran yang fleksibel sesuai dengan preferensi mereka, baik melalui *transfer* bank atau *e-wallet*. Ini memberikan kenyamanan dan mempermudah proses pembayaran bagi pengguna dengan berbagai pilihan.

2. Transparansi Pembayaran:

Dengan menampilkan ringkasan pesanan yang mencakup layanan, harga, dan alamat, pengguna dapat memverifikasi kembali informasi sebelum melanjutkan ke pembayaran. Ini mengurangi kemungkinan kesalahan dan memastikan bahwa pengguna mengetahui dengan pasti apa yang mereka bayar.

3. Simulasi yang Berguna:

Fitur simulasi pembayaran memberikan kenyamanan bagi pengguna, terutama ketika mereka hanya ingin melihat bagaimana proses pembayaran bekerja tanpa benar-benar melakukan pembayaran. Ini berguna dalam tahap pengujian atau uji coba sistem.

4. Integrasi yang Aman:

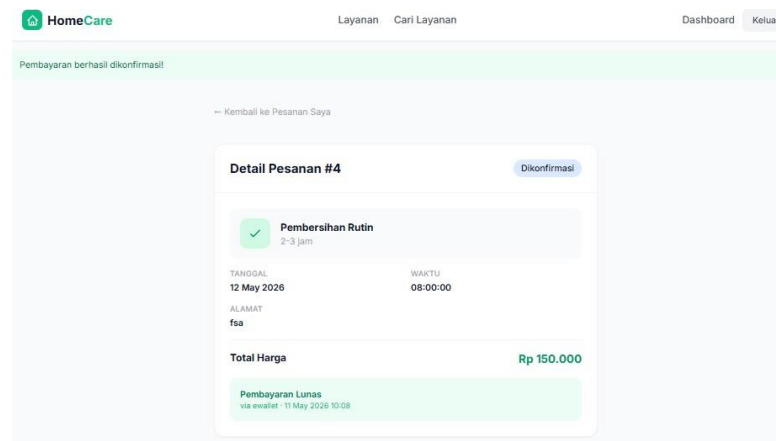
Mengintegrasikan sistem pembayaran dengan *payment gateway* yang aman menjamin proses pembayaran yang cepat, efisien, dan terpercaya. Ini meningkatkan kepercayaan pengguna terhadap aplikasi karena mereka tahu bahwa data pembayaran mereka diproses dengan cara yang aman.

Evaluasi Sistem Pembayaran Secara keseluruhan, sistem pembayaran ini dirancang untuk mempermudah pengguna dalam menyelesaikan transaksi pemesanan layanan *home care*. Metode pembayaran yang fleksibel, ditambah dengan fitur simulasi pembayaran dan konfirmasi pesanan, membuat pengguna merasa nyaman dan aman dalam menggunakan layanan ini.

Integrasi dengan *payment gateway* memastikan bahwa transaksi pembayaran dapat dilakukan dengan aman dan efisien.

4.8. Halaman Konfirmasi Pembayaran dan Detail Pesanan

Halaman Konfirmasi Pembayaran yang ditampilkan di sini menunjukkan status pesanan setelah proses pembayaran berhasil. Setelah pengguna menyelesaikan pembayaran dan memilih metode pembayaran yang sesuai, halaman ini menyajikan rincian akhir dari pesanan yang telah dibayar, termasuk konfirmasi pembayaran dan informasi pesanan yang telah terverifikasi.



Gambar 4. 8 Konfirmasi Pembayaran

Penjelasan Detail Halaman Konfirmasi Pembayaran

1. Pesan Konfirmasi Pembayaran

Pesan Berhasil: Di bagian atas halaman, terdapat pesan berwarna hijau yang berbunyi "Pembayaran berhasil dikonfirmasi!". Pesan ini memberikan *feedback* langsung kepada pengguna bahwa transaksi pembayaran mereka telah diterima dan diproses dengan sukses. Ini memberi rasa aman dan kepastian kepada pengguna bahwa pembayaran mereka telah berhasil.

2. Detail Pesanan

Nomor Pesanan: Halaman ini menampilkan Detail Pesanan dengan nomor pesanan yang unik, yaitu #4. Nomor ini membantu pengguna untuk melacak pesanan mereka dengan lebih mudah.

Layanan yang Dipesan: Di bawah nomor pesanan, rincian layanan yang dipilih oleh pengguna ditampilkan, yaitu "Pembersihan Rutin" dengan estimasi waktu 2-3 jam. Ini memastikan bahwa pengguna tahu persis layanan yang telah mereka pilih dan bayar.

Tanggal dan Waktu Layanan: Tanggal dan waktu yang telah dipilih oleh pengguna untuk layanan ini adalah 12 May 2026 - 08:00:00. Ini mengingatkan pengguna mengenai kapan layanan mereka akan dilakukan.

Alamat Layanan: Alamat tempat layanan akan dilakukan juga ditampilkan di sini, dalam contoh ini adalah "fsa". Menampilkan alamat ini memastikan bahwa semua informasi terkait lokasi telah tercatat dengan benar untuk layanan yang akan dilaksanakan.

Total Harga: Harga total untuk layanan ini yang telah dibayar adalah Rp 150.000. Ini memastikan bahwa pengguna mengetahui biaya yang telah mereka bayar untuk layanan tersebut.

3. Status Pembayaran

Pembayaran Lunas: Di bagian bawah detail pesanan, tertera status "Pembayaran Lunas", menunjukkan bahwa pembayaran untuk layanan ini telah berhasil dilakukan. Waktu dan metode pembayaran juga ditampilkan, dalam hal ini via *e-wallet* pada 11 May 2026 pukul 10:08. Ini memberikan pengguna kejelasan tentang waktu dan cara pembayaran yang mereka pilih.

4. Tombol Kembali ke Pesanan Saya

Di bagian atas halaman, terdapat tombol "Kembali ke Pesanan Saya" yang memungkinkan pengguna untuk kembali ke halaman utama pesanan saya untuk melihat riwayat pesanan lain atau melakukan pengecekan lebih lanjut mengenai pesanan yang telah mereka buat.

Proses Pengolahan Data Pembayaran dan Konfirmasi Pesanan

1. Pengolahan Pembayaran:

Setelah pengguna menyelesaikan pembayaran melalui salah satu metode yang tersedia, sistem memverifikasi pembayaran tersebut dengan *payment gateway* yang terintegrasi. Setelah pembayaran berhasil, status pesanan diubah menjadi "Lunas", dan pengguna menerima notifikasi di halaman ini.

2. Integrasi dengan Sistem Pembayaran:

Setelah pembayaran dikonfirmasi, data terkait transaksi, seperti waktu pembayaran, metode pembayaran, dan status pesanan, disimpan dalam *database MySQL*. Sistem ini memastikan bahwa informasi pembayaran terkait pesanan tetap konsisten dan mudah dilacak.

3. Pembaruan Status Pesanan:

Ketika pembayaran berhasil, sistem secara otomatis memperbarui status pesanan menjadi "Lunas", yang memberi tahu pengguna bahwa layanan akan diproses dan dijadwalkan sesuai dengan rincian yang telah diberikan.

Keuntungan Halaman Konfirmasi Pembayaran

1. Transparansi Pembayaran:

Dengan menampilkan pesan konfirmasi pembayaran dan informasi rinci tentang pesanan yang telah dibayar, pengguna dapat dengan mudah memverifikasi bahwa transaksi mereka telah berhasil dan diproses dengan benar. Ini meningkatkan kepercayaan pengguna terhadap sistem.

2. Pemberitahuan yang Jelas:

Menyediakan pesan berhasil dan status pembayaran memberikan kepastian kepada pengguna bahwa tidak ada masalah dengan pembayaran mereka. Ini membantu mencegah kebingungan atau pertanyaan terkait status transaksi.

3. Fungsionalitas Pengelolaan Pesanan:

Fitur "Kembali ke Pesanan Saya" memungkinkan pengguna untuk mengakses riwayat pesanan mereka, memberi mereka kontrol lebih atas informasi yang mereka butuhkan tanpa harus mencari secara manual.

4. Efisiensi Proses Pembayaran:

Sistem pembayaran yang terintegrasi dengan payment *gateway* memastikan bahwa proses pembayaran berjalan dengan lancar dan efisien, memungkinkan pengguna untuk segera melanjutkan ke tahap berikutnya tanpa penundaan.