

BAB II

KAJIAN PUSTAKA

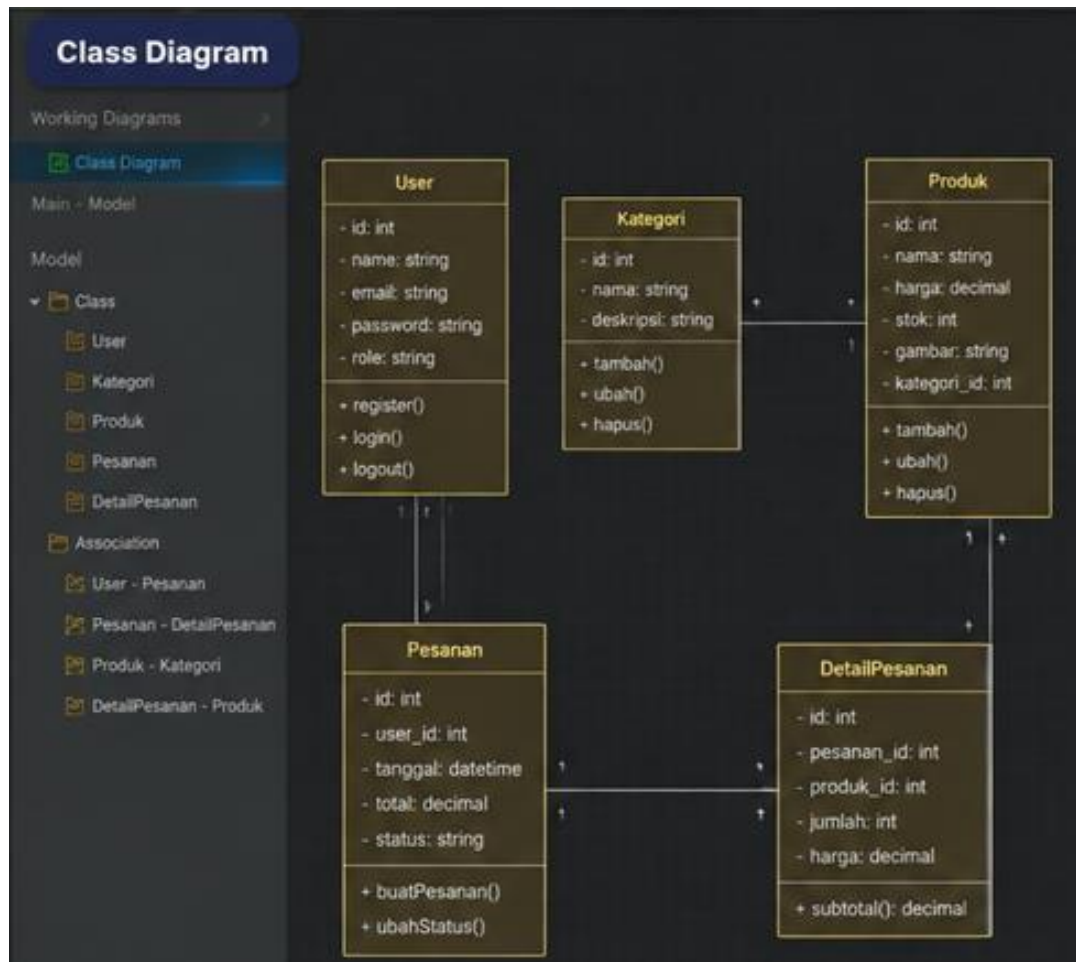
2.1. *Sistem informasi*

secara umum dipahami sebagai spesifikasi konseptual yang eksplisit untuk suatu domain, di mana konsep-konsep utama (misalnya Layanan, Area/Ruang, Paket, Alat/Bahan, Durasi, Harga, Pemesanan) didefinisikan secara jelas beserta relasi antar konsepnya. Pendekatan ini membantu mengurangi ambiguitas istilah dan meningkatkan konsistensi pemahaman domain, karena menyatukan kosakata dan struktur hubungan yang digunakan bersama oleh pengguna maupun sistem. Dalam konteks penelitian ini, digunakan untuk memodelkan layanan *home care* kebersihan/perawatan rumah agar informasi layanan (cakupan pekerjaan, area yang dikerjakan, kebutuhan alat/bahan, dan aturan paket) dapat direpresentasikan secara terstruktur dan dapat diproses mesin. [1]

Selain sebagai “kamus istilah”, juga memuat batasan/aturan konseptual (misalnya hubungan layanan membutuhkan alat, paket terdiri dari beberapa layanan, atau layanan cocok untuk area tertentu).

Dengan begitu, sistem tidak hanya menyimpan data layanan, tetapi juga memahami keterkaitan antar data sehingga mendukung penelusuran informasi secara lebih semantik (berdasarkan makna dan relasi), bukan sekadar pencocokan kata. Panduan pengembangan yang umum dipakai menekankan pentingnya penentuan ruang lingkup, definisi kelas dan properti, serta konsistensi penamaan agar mudah dipelihara dan dikembangkan. [2] [5]

representasi formal dari pengetahuan dalam domain tertentu. *Home Care* digunakan untuk memodelkan entitas seperti Paket Layanan, Layanan, Area Rumah, Alat, dan Bahan beserta relasinya.



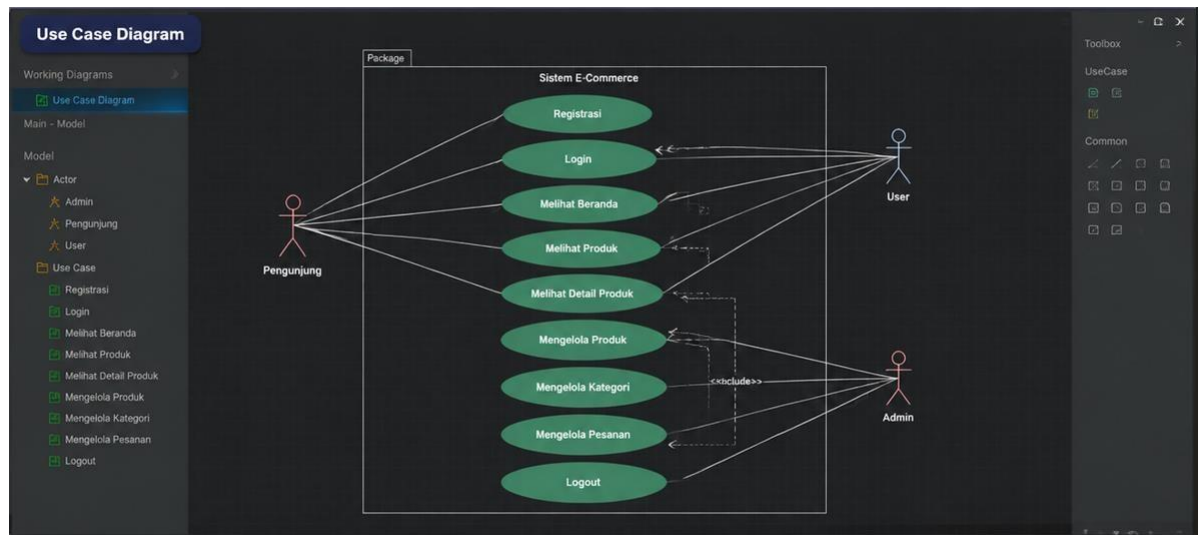
Gambar 2. 1 Struktur Home Care

2.2. Teori Penunjang Penelitian

2.2.1. Competency Questions

CQs digunakan untuk mengukur apakah dapat menjawab kebutuhan domain. Contoh CQs *Home Care*:

1. Layanan apa saja yang tersedia di dapur?
2. Paket layanan apa yang mencakup layanan tertentu?
3. Alat/bahan apa yang dibutuhkan untuk layanan tertentu.



Gambar 2. 2 Use Case Diagram Home Care

User / Pelanggan Merupakan aktor utama yang menggunakan *website* untuk memesan layanan *Home Care*.

Hak dan fungsinya:

1. Membuka *website (Homepage)*
2. Melihat daftar produk / layanan
3. Melihat detail produk
4. Menambahkan layanan ke keranjang
5. *Checkout* dan konfirmasi pesanan

Admin Merupakan aktor yang mengelola sistem di *backend*.

Hak dan fungsinya:

1. Menambah, mengubah, dan menghapus paket layanan
2. Mengelola kategori layanan

3. Mengelola pesanan (memverifikasi, memproses, menyelesaikan)

Diagram ini menunjukkan interaksi kedua aktor dengan sistem. *User* fokus pada pemesanan layanan, sementara *Admin* fokus pada pengelolaan sistem dan layanan.

Dalam penelitian “layanan *home care* kebersihan rumah”, tahapan rekayasa membuat proses perancangan menjadi lebih rapi: peneliti dapat mengumpulkan istilah layanan (misalnya *general cleaning*, *deep cleaning*, cuci sofa, cuci karpet), mengelompokkan ke konsep inti (kelas), menentukan relasi (misalnya Layanan-mengerjakan Area, Layanan membutuhkanAlat/Bahan),

lalu menambahkan aturan sederhana (misalnya paket minimal mencakup sejumlah layanan tertentu). Dengan pendekatan metodologis, lebih siap digunakan sebagai *knowledge base* untuk prototipe sistem informasi layanan. [3]

2.2.2. Semantic Web: RDF, OWL, dan SPARQL

Untuk membuat dapat dipakai sistem secara standar dan *interoperabel*, penelitian ini memanfaatkan teknologi *Semantic Web*. *RDF* (*Resource Description Framework*) menyediakan model data berbasis *graf* (*triple* subjek–predikat–objek) yang cocok untuk merepresentasikan relasi antarkonsep (misalnya LayananA membutuhkan AlatX, PaketY terdiri dari LayananA). *RDF* juga mendukung pengorganisasian graf melalui *named graph* dan *dataset* sehingga pengelolaan pengetahuan dapat lebih terstruktur. [5]

Di atas *RDF*, *OWL* (*Web Ontology Language*) digunakan untuk memodelkan secara formal, termasuk definisi *class*, *property*, individual, serta batasan semantik (misalnya kardinalitas atau relasi tertentu). *OWL* juga mendukung *reasoning* (penalaran) sehingga sistem dapat memeriksa konsistensi dan menyimpulkan fakta turunan berdasarkan aturan/relasi yang didefinisikan. Dalam implementasi, *OWL* umumnya dipertukarkan sebagai dokumen *RDF*. [4]

Setelah pengetahuan direpresentasikan dalam *RDF/OWL*, *SPARQL* digunakan sebagai bahasa kueri untuk mengambil informasi dari graf *RDF*. *SPARQL* mendukung pencarian pola graf (*graph pattern*), termasuk pola wajib dan opsional, agregasi, filter, hingga *subquery*. Dalam penelitian ini, *SPARQL* dipakai untuk menguji apakah dapat menjawab kebutuhan informasi pelanggan, misalnya: “layanan apa yang cocok untuk area dapur?”, “paket apa yang mencakup *deep cleaning*?”, atau “alat/bahan apa yang dibutuhkan untuk cuci sofa?”. [6]

Semantic Web merupakan sebuah visi dan pengembangan dari *World Wide Web (WWW)* yang diinisiasi oleh *W3C (World Wide Web Consortium)*. Pada *web* konvensional (*Web 2.0*), informasi disajikan dalam format yang utamanya dirancang untuk dibaca oleh manusia (*human-readable*), seperti *HTML* dan *CSS*.

Namun, komputer atau mesin pencari sering kali tidak memahami makna logis (*semantic*) dari kata-kata tersebut, melainkan hanya membacanya sebagai untaian karakter (*string*) biasa. *Semantic Web* hadir untuk mengatasi keterbatasan ini dengan memberikan lapisan metadata yang terstruktur sehingga data tidak hanya dapat ditampilkan, tetapi juga dapat dipahami, dihubungkan, dan diproses secara otomatis oleh mesin (*machine-readable*).

Untuk mewujudkan arsitektur *Semantic Web* yang utuh, terdapat beberapa lapisan teknologi (*Semantic Web Layer Cake*) yang saling mendukung. Komponen paling mendasar dan krusial dalam pemodelan data pada arsitektur ini adalah *Resource Description Framework (RDF)*.

1. Konsep Dasar *RDF (Resource Description Framework)*

RDF adalah sebuah standar *framework* yang digunakan untuk merepresentasikan informasi mengenai sumber daya (*resources*) yang ada di *web*. Karakteristik utama dari *RDF* adalah kemampuannya untuk mendeskripsikan data tanpa terikat oleh skema basis data yang kaku seperti pada basis data relasional (*RDBMS*). *RDF* menggunakan konsep pengidentifikasi global yang disebut *Uniform Resource Identifier (URI)* atau *Internationalized Resource Identifier (IRI)*.

Penggunaan *URI* memastikan bahwa setiap entitas, properti, maupun konsep di dalam jaringan *Semantic Web* memiliki identitas unik yang dapat diakses secara global, sehingga menghindari ambiguitas data.

2. Struktur *RDF Triples*

Data di dalam *RDF* dimodelkan dalam bentuk grafik berarah (*directed graph*) yang tersusun atas pernyataan tiga serangkai, atau yang secara formal disebut sebagai *RDF Triples*. Struktur tiga serangkai ini merepresentasikan sebuah fakta logis yang terdiri dari tiga komponen utama, yaitu:

Subject: Entitas, objek, atau *resource* utama yang sedang dideskripsikan. *Subject* harus direpresentasikan menggunakan *URI/IRI* agar memiliki identitas unik di *web*, atau menggunakan *blank node* untuk entitas anonim.

1. *Predicate*: Properti, atribut, atau karakteristik yang menyatakan jenis hubungan (relasi) antara *Subject* dan *Object*. *Predicate* juga wajib berupa *URI/IRI* yang merujuk pada kosakata (*vocabulary*) atau tertentu.
2. *Object*: Nilai dari properti yang dinyatakan oleh *Predicate*. *Object* dapat berupa dua hal: sebuah *resource* lain (berupa *URI/IRI*) yang membentuk relasi antar entitas, atau berupa data *literal* (nilai primitif seperti *teks string*, angka integer, tanggal, atau boolean) yang disertai dengan tipe data tertentu (*datatype literal*).

Untuk memberikan gambaran konkret mengenai implementasi *RDF Triples* pada sistem *home cleaning service* yang dibangun dalam Tugas Akhir ini, berikut adalah representasi logis dan matematis dari beberapa data operasional:

Contoh Pernyataan Relasi Antar Entitas (*Object Property*):

1. *Subject*: http://homecleaning.org/ns#Layanan_DeepCleaning
2. *Predicate*: <http://homecleaning.org/ns#membutuhkanAlat>
3. *Object*: http://homecleaning.org/ns#Vacuum_Cleaner

Penjelasan Logis: Pernyataan ini menegaskan secara semantik bahwa instansiasi layanan *Deep Cleaning* memiliki hubungan ketergantungan fisik dengan alat *Vacuum Cleaner*. Mesin pencari atau agen cerdas dapat menyimpulkan bahwa jika seorang pelanggan memesan *Deep Cleaning*, maka sistem harus memastikan ketersediaan *Vacuum Cleaner*.

Contoh Pernyataan Atribut Nilai (*Data Property*):

1. *Subject*: http://homecleaning.org/ns#Layanan_DeepCleaning

2. *Predicate*: `http://homecleaning.org/ns#memilikiTarif`
3. *Object*: `"150000"^^xsd:integer`

Penjelasan Logis: Pernyataan ini menghubungkan entitas layanan dengan nilai literal angka 150000 yang secara eksplisit didefinisikan bertipe data *integer* menggunakan *XML Schema Definition* (`xsd:integer`).

3. *Graf RDF (RDF Graph)* dan Serialisasi

Kumpulan dari banyak *RDF Triples* yang saling terhubung satu sama lain akan membentuk sebuah jaringan raksasa yang disebut *Graf RDF (RDF Graph)*. Dalam graf ini, *Subject* dan *Object* bertindak sebagai simpul (*nodes*),

sedangkan *Predicate* bertindak sebagai garis panah berarah (*directed edges*) yang menghubungkan simpul-simpul tersebut. Melalui struktur graf inilah fitur pencarian semantik dapat melakukan penelusuran hubungan (*relationship reasoning*) yang tidak bisa dilakukan secara instan oleh kueri *JOIN* konvensional di *database SQL*.

Dalam implementasi teknisnya, *graf RDF* ini disimpan dan ditulis ke dalam berkas fisik menggunakan beberapa format penulisan standar (serialisasi) yang diakui oleh W3C, antara lain:

1. *RDF/XML*: Format berbasis *XML* yang merupakan standar awal *RDF*.

Format ini sangat terstruktur namun cenderung sulit dibaca secara langsung oleh manusia karena sintaksisnya yang padat.

2. *Turtle (Terse RDF Triple Language)*:

Format serialisasi yang paling populer, ringkas, dan sangat mudah dibaca baik oleh manusia maupun mesin. *Turtle* menggunakan singkatan-singkatan sintaksis yang efisien untuk menuliskan graf *RDF*.

3. *JSON-LD (JSON for Linking Data):*

Format berbasis *JSON* yang sangat ramah bagi pengembang *web* (*web developers*), khususnya dalam lingkungan aplikasi modern seperti *framework Laravel*, karena strukturnya dapat dipetakan langsung ke dalam objek *array PHP* atau *JavaScript*.

Pada penelitian Tugas Akhir ini, pemodelan data semantik untuk domain layanan kebersihan rumah diwujudkan menggunakan format serialisasi *Turtle (.ttl)* dan *OWL/XML (.owl)* guna memastikan performa pembacaan data yang optimal saat diproses oleh *library EasyRdf* di dalam sistem *Laravel*.

2.2.3. Competency Questions (CQs) sebagai Kebutuhan dan Evaluasi

Salah satu tantangan dalam penelitian Sistem Informasi adalah memastikan yang dibangun benar-benar relevan dengan kebutuhan domain, bukan sekadar “diagram konsep” yang bagus secara teori. Karena itu, pendekatan *Competency Questions (CQs)* digunakan untuk menyatakan daftar pertanyaan yang “wajib bisa dijawab” oleh. *CQs* membantu mendefinisikan ruang lingkup Sistem Informasi secara terukur dan mengarahkan peneliti saat memilih konsep, relasi, dan batasan. [7]

Dalam domain *home care* kebersihan rumah, *CQs* dapat berupa: layanan berdasarkan area tertentu, kombinasi layanan dalam paket, kebutuhan alat/bahan per layanan, atau aturan sederhana (misalnya layanan tertentu tidak dapat digabungkan dengan kondisi tertentu). Setelah diimplementasikan, *CQs* dapat diuji secara praktis menggunakan kueri *SPARQL*; jika pertanyaan tidak dapat dijawab, berarti perlu direvisi (menambah konsep/relasi atau memperbaiki definisi). [7]

2.2.4. Tools Pengembangan: Protégé

Dalam praktiknya, pengembangan memerlukan alat bantu agar proses definisi kelas, properti, dan individu lebih mudah serta dapat divalidasi. *Protégé* merupakan editor yang populer dan luas dipakai untuk membangun *OWL*, mengelola struktur kelas/properti, serta mendukung berbagai *plugin* untuk visualisasi maupun *reasoning*. *Protégé* juga banyak digunakan dalam pengembangan sistem berbasis pengetahuan karena sifatnya yang fleksibel dan didukung komunitas luas. [8], [9]

Dalam penelitian ini, *Protégé* dapat digunakan untuk: (1) membuat struktur layanan, area, paket, dan alat/bahan, (2) memasukkan contoh individu layanan (misalnya *DeepCleaning*, *CuciSofa*) beserta atributnya, dan (3) mengeksport ke format *OWL/RDF* agar dapat diuji menggunakan *SPARQL* pada lingkungan *RDF store* atau *query engine*. [9]

2.2.5. UML sebagai Notasi Perancangan Sistem

Walaupun inti penelitian adalah, dokumentasi sistem tetap memerlukan notasi pemodelan yang jelas agar rancangan mudah dipahami oleh dosen/pembaca dan dapat dijadikan acuan implementasi prototipe. *UML (Unified Modeling Language)* merupakan standar notasi pemodelan untuk memvisualisasi, menspesifikasi, dan mendokumentasikan artefak sistem.

Dalam penelitian ini, *UML* dapat digunakan untuk menjelaskan aktor dan kebutuhan fungsional prototipe (misalnya Pelanggan melihat katalog layanan/paket dan *Admin* mengelola layanan), serta untuk memvisualisasi struktur konseptual yang beririsan dengan (misalnya *Class Diagram* untuk menggambar konsep layanan-area-alat). [10]

Penggunaan *UML* membantu menjadi lebih “informatika banget” karena terlihat jelas: (1) apa yang dibangun (kebutuhan sistem), dan (2) bagaimana struktur konsepnya (yang kemudian dipetakan ke). Selain itu, *UML* memudahkan pembaca membedakan bagian “model pengetahuan ” dengan bagian “prototipe sistem informasi yang memanfaatkan”. [10]

Evaluasi dilakukan dengan pendekatan *task-based* dan *semiotic metrics* untuk memastikan mencakup seluruh domain dan dapat digunakan di *website*.

Website menggunakan *Laravel 10* sebagai *backend*, *TailwindCSS* + *AlpineJS* untuk *frontend*, serta integrasi *RDF/TTL* untuk. Sistem ini memungkinkan pelanggan melihat paket layanan, estimasi biaya, dan durasi secara *real-time*.

2.2.6. Penelitian Terdahulu

Untuk menguatkan konteks *domain*, penelitian terdahulu tentang *platform* layanan rumah tangga menunjukkan bahwa sektor *household services* menghadapi tantangan seperti kebutuhan kecepatan, reliabilitas, dan pemilihan penyedia layanan yang tepat; hal ini menuntut sistem informasi yang mampu menyajikan layanan secara jelas dan mudah ditelusuri. [11]

Selain itu, studi terkait sistem pemesanan *home cleaning service* berbasis *web* menunjukkan adanya kebutuhan pengelolaan layanan, jadwal, dan detail harga secara lebih terstruktur untuk meningkatkan kemudahan pemesanan dan pengolahan layanan oleh *admin* ketika terjadi perubahan layanan. Temuan seperti ini sejalan dengan motivasi penelitian: menyusun struktur pengetahuan layanan yang konsisten agar informasi layanan dapat disajikan dan dikelola dengan lebih rapi. [12]

Penelitian terdahulu berfungsi sebagai landasan komparatif untuk menegaskan posisi, kontribusi, serta pembeda (*gap analysis*) dari penelitian yang dilakukan saat ini. Penelitian mengenai integrasi *Semantic Web* dan sistem informasi berbasis *web* telah banyak dikembangkan, namun penerapan spesifik pada pencarian semantik layanan *home cleaning* menggunakan *framework Laravel* masih memiliki ruang pengembangan yang luas. Berikut adalah beberapa penelitian relevan yang menjadi acuan:

Penelitian oleh Pratama et al. (2022) mengembangkan sistem pencarian berbasis semantik untuk komoditas pertanian menggunakan dan *Apache Jena*. Penelitian tersebut berhasil membuktikan bahwa pencarian semantik mampu menghasilkan relevansi data yang lebih tinggi dibandingkan dengan kueri *SQL* konvensional (*keyword-based*). Namun, penelitian tersebut belum mengintegrasikan sistem dengan *framework web* modern dan berfokus pada arsitektur monolitik.

Penelitian oleh Wijaya & Santoso (2023) membangun *domain* untuk *e-commerce* jasa lokal. Fokus utama mereka adalah memodelkan relasi antara penyedia jasa (*vendor*) dan konsumen. Meskipun pemodelan cukup detail, pengujian sistem hanya terbatas pada aplikasi *desktop* protegenya saja dan belum diimplementasikan ke dalam aplikasi berbasis *web* yang siap pakai oleh pengguna akhir (*end-user*).

Penelitian oleh Utami (2024) menerapkan *framework Laravel* untuk aplikasi *marketplace* pembersihan rumah, namun basis data yang digunakan sepenuhnya masih berupa basis data relasional (*MySQL*). Akibatnya, sistem mengalami keterbatasan ketika menangani pencarian layanan yang bersifat implisit atau membutuhkan penalaran hubungan relasional antar entitas jasa (seperti kecocokan alat dan jenis ruangan).

Berdasarkan literatur di atas, penelitian Tugas Akhir ini mengisi celah (gap) tersebut dengan cara mengintegrasikan *framework Laravel* sebagai antarmuka dan pengolah logika bisnis dengan *Triple Store* berbasis *OWL*. Domain yang dipilih secara spesifik adalah *home cleaning service*, yang memodelkan relasi kompleks antara pelanggan, jenis layanan, ruangan, alat kebersihan, serta tenaga penyedia jasa secara semantik.

Untuk mempermudah pemahaman mengenai posisi penelitian ini dibandingkan dengan penelitian terdahulu, berikut disajikan tabel komparasi pada Tabel 2.1

Tabel 2. 1 Penelitian Terdahulu

No	Peneliti&Tahun	Metode/Teknologi	Domain Penelitian	Kelebihan	Keterbatasan/Gap
1	Pratama et al. (2022)	<i>Apache Jena, SPARQL</i>	Komoditas Pertanian	Relevansi pencarian semantik sangat tinggi.	Arsitektur masih monolitik, belum menggunakan <i>framework</i> modern.
2	Wijaya & Santoso (2023)	<i>Protégé, OWL, Jasa</i>	Marketplace Jasa Lokal	Pemodelan taksonomi kelas sangat mendalam.	Implementasi terbatas pada <i>Protégé</i> , belum berbasis web <i>real-time</i> .
3	Utami (2024)	<i>Laravel, MySQL, Relasional</i>	Home Cleaning Service	Sistem berbasis web stabil, UX/UI matang dengan <i>Laravel</i> .	Pencarian masih kaku (<i>keyword-based</i>), tidak ada <i>relationship reasoning</i> .
4	Alvin (2026)	<i>Laravel, PHP (EasyRdf), OWL, SPARQL, MySQL</i>	Semantics-based Home Cleaning Service	Menggabungkan performa web <i>Laravel</i> dengan kecerdasan relasional <i>OWL</i> via <i>SPARQL</i> .	Fokus pada optimalisasi kueri semantik untuk pencarian layanan kebersihan.

Berdasarkan Tabel 2.1 di atas, dapat ditarik kesimpulan bahwa pemanfaatan teknologi *Semantic Web*, telah terbukti mampu meningkatkan akurasi dan relevansi pencarian dibandingkan metode pencarian konvensional yang hanya berbasis pencocokan kata kunci

(*keyword-based*). Namun, implementasi pada penelitian-penelitian sebelumnya masih memiliki keterbatasan pada sisi arsitektur sistem. Sistem yang dibangun umumnya masih bersifat monolitik atau sebatas purwarupa (*prototype*) pada perangkat lunak *desktop* seperti *Protégé*, sehingga kurang adaptif jika diimplementasikan sebagai aplikasi berbasis *web* yang diakses oleh banyak pengguna secara *real-time*.

Di sisi lain, penelitian yang telah memanfaatkan *framework web* modern seperti *Laravel* masih sangat bergantung pada basis data relasional (*MySQL*) sepenuhnya. Pendekatan ini memiliki kelemahan mendasar ketika sistem dituntut untuk menampilkan rekomendasi layanan berdasarkan hubungan antar entitas yang implisit dan kompleks—misalnya, menghubungkan keahlian spesifik pekerja, alat yang dibutuhkan, dan spesifikasi area pengerjaan. Basis data relasional akan membutuhkan operasi kueri *JOIN* yang sangat panjang dan membebani kinerja peladen (*server*) ketika relasi data semakin dalam.

Berangkat dari celah penelitian (*research gap*) tersebut, Tugas Akhir ini mengusulkan sebuah pembaruan (*novelty*) dengan cara mengawinkan keunggulan dari kedua pendekatan di atas. Penelitian ini tetap menggunakan *framework Laravel* dan *MySQL* untuk menangani kecepatan transaksi data operasional dasar (seperti autentikasi pengguna dan pencatatan transaksi), namun mendelegasikan tugas pencarian dan rekomendasi layanan kebersihan ke dalam mesin penalaran semantik (*Semantic Reasoning Engine*) menggunakan bahasa kueri *SPARQL*.

Melalui pendekatan arsitektur hibrida (*hybrid*) ini, aplikasi pencarian layanan *home cleaning service* tidak hanya memiliki antarmuka yang ramah pengguna (*user-friendly*) dan kinerja yang stabil, tetapi juga memiliki "kecerdasan" relasional dalam memahami konteks pencarian pengguna karena didukung oleh struktur *OWL* yang kaya akan relasi logis.

Lebih lanjut, urgensi penerapan teknologi *Semantic Web* pada sistem ini didorong oleh karakteristik domain layanan *home cleaning* yang sangat dinamis dan saling bergantung (*interdependent*). Dalam operasional di dunia nyata, kebutuhan pembersihan sebuah area (misalnya kamar mandi atau ruang tamu) sangat dipengaruhi oleh tingkat kekotoran, yang pada gilirannya menentukan jenis alat (seperti *vacuum cleaner* basah atau kering) serta bahan kimia pembersih yang harus dibawa. Pada sistem konvensional, setiap variasi kombinasi ini harus direkam satu per satu ke dalam tabel basis data yang kaku, sehingga memicu terjadinya

redundansi data dan kesulitan dalam pemeliharaan sistem (*system maintenance*) ketika ada penambahan layanan baru.

Dengan mengadopsi *OWL*, sistem yang diusulkan dalam penelitian ini mampu merepresentasikan pengetahuan tersebut layaknya cara manusia berpikir. Sistem tidak lagi melihat "Area", "Layanan", dan "Pekerja" sebagai entitas yang berdiri sendiri, melainkan sebagai sebuah graf terhubung.

"Untuk mengatasi keterbatasan arsitektur basis data relasional konvensional tersebut, diperlukan sebuah pendekatan pemodelan data yang lebih dinamis dan terstruktur. Pendekatan ini harus mampu menangkap makna kontekstual serta relasi kompleks antar elemen operasional tanpa mengharuskan perombakan skema tabel secara besar-besaran setiap kali terjadi perubahan. Oleh karena itu, pengintegrasian basis pengetahuan berbasis graf (*graph-based knowledge base*) menjadi solusi yang sangat relevan untuk menjembatani celah (gap) penelitian tersebut."

direkam satu per satu ke dalam tabel basis data yang kaku, sehingga memicu terjadinya redundansi data dan kesulitan dalam pemeliharaan sistem (*system maintenance*) ketika ada penambahan layanan baru.

Untuk mengatasi keterbatasan arsitektur basis data relasional konvensional tersebut, diperlukan sebuah pendekatan pemodelan data yang lebih dinamis dan terstruktur. Pendekatan ini harus mampu menangkap makna kontekstual serta relasi kompleks antar elemen operasional tanpa mengharuskan perombakan skema tabel secara besar-besaran setiap kali terjadi perubahan. Oleh karena itu, pengintegrasian basis pengetahuan berbasis graf (*graph-based knowledge base*) menjadi solusi yang sangat relevan untuk menjembatani celah (gap) penelitian tersebut.

Dengan mengadopsi standar pemodelan *OWL* sistem yang diusulkan dalam penelitian ini mampu merepresentasikan pengetahuan tersebut layaknya cara manusia berpikir. Sistem tidak lagi melihat "Area", "Layanan", dan "Pekerja" sebagai entitas yang berdiri sendiri, melainkan sebagai sebuah graf terhubung."