KERJA PRAKTIK - EF234603

Perancangan dan Implementasi Aplikasi Web Manajemen *Sales Agent* dengan Gamifikasi dan *Leaderboard* PT. Telkom Indonesia Regional V

PT. Telekomunikasi Selular, Telkom Landmark Tower,
Jl. Dr. Ir. H. Soekarno No.175, Klampis Ngasem, Kec. Sukolilo,
Surabaya, Jawa Timur 60116
Periode: 30 Juni 2025-30 September 2025

Oleh:

Helsa Sriprameswari Putri 5025221154
Aurelia Natasya Putri 5025221237

Pembimbing Departemen
Dr. Yudhi Purwananto, S.Kom., M.Kom.

Pembimbing Lapangan
Raden Muhammad Rizal Ghazali, S.M., M.B.A.

DEPARTEMEN TEKNIK INFORMATIKA
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2025

KERJA PRAKTIK – EF234603

Perancangan dan Implementasi Aplikasi Web Manajemen *Sales Agent* dengan Gamifikasi dan *Leaderboard* PT. Telkom Indonesia Regional V

PT. Telekomunikasi Selular, Telkom Landmark Tower,
Jl. Dr. Ir. H. Soekarno No.175, Klampis Ngasem, Kec. Sukolilo,
Surabaya, Jawa Timur 60116
Periode: 30 Juni 2025-30 September 2025

Oleh:

Helsa Sriprameswari Putri 5025221154
Aurelia Natasya Putri 5025221237

Pembimbing Departemen
Dr. Yudhi Purwananto, S.Kom., M.Kom.

Pembimbing Lapangan
Raden Muhammad Rizal Ghazali, S.M., M.B.A.

DEPARTEMEN TEKNIK INFORMATIKA
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember
Surabaya 2025

[Halaman ini sengaja dikosongkan]

DAFTAR ISI

[Halaman ini sengaja dikosongkan]

DAFTAR GAMBAR

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

KERJA PRAKTIK

Perancangan dan Implementasi Aplikasi Web Manajemen *Sales Agent* dengan Gamifikasi dan *Leaderboard* PT. Telkom Indonesia Regional V

Oleh:

Helsa Sriprameswari Putri	5025221154
Aurelia Natasya Putri	5025221237

Disetujui oleh Pembimbing Kerja Praktik:

1. Dr. Yudhi Purwananto, S.Kom., M.Kom
NIP. 197007141997031002

2. Raden Muhammad Rizal Ghazali, S.M., M.B.A.

(Pembimbing Departemen)

Signed by RADEN MUHAMAD RIZAL GHAZALI (RMR6088)
Signed on Sep 30, 2022 [illegible]

(Pembimbing Lapangan)

[Halaman ini sengaja dikosongkan]

Perancangan dan Implementasi Aplikasi Web Manajemen *Sales Agent* dengan Gamifikasi dan *Leaderboard* PT. Telkom Indonesia Regional V

Nama Mahasiswa	: Helsa Sriprameswari Putri
NRP	: 5025221154
Nama Mahasiswa	: Aurelia Natasya Putri
NRP	: 5025221237
Departemen	: Teknik Informatika FTEIC-ITS
Pembimbing Departemen	: Dr. Yudhi Purwananto, S.Kom, M.Kom.
Pembimbing Lapangan	: Raden Muhammad Rizal Ghazali

ABSTRAK

PT Telkom Indonesia Regional V merupakan perusahaan di bidang telekomunikasi dan layanan digital dengan berbagai produk seperti HSI, WMS, Netmonk, OCA, Pijar, dan Eazy. Selama kerja praktik, kami mengembangkan aplikasi web sales agent management untuk memantau dan mengelola kinerja tenaga penjual secara terpusat.

Aplikasi berbasis Laravel dengan arsitektur MVC ini memiliki fitur pengelolaan data sales agent, leaderboard, serta gamifikasi berupa poin dan badge. Sistem menggunakan PostgreSQL, Tailwind CSS, template Konrix, dan Git sebagai version control. Implementasi menunjukkan aplikasi berjalan baik dan membantu perusahaan memantau kinerja serta mendorong pencapaian target penjualan.

Kata Kunci: *Website, Sales Agent, Gamifikasi, Leaderboard, Laravel, PostgreSQL*

[Halaman ini sengaja dikosongkan]

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Allah SWT atas penyertaan dan karunia-Nya sehingga penulis dapat menyelesaikan salah satu kewajiban penulis sebagai mahasiswa Departemen Teknik Informatika ITS yaitu Kerja Praktik yang berjudul: Perancangan dan Implementasi Aplikasi Web Manajemen Sales Agent dengan Gamifikasi dan *Leaderboard* PT Telkom Indonesia Regional V.

Penulis menyadari bahwa masih banyak kekurangan baik dalam melaksanakan kerja praktik maupun penyusunan buku laporan kerja praktik ini. Namun penulis berharap buku laporan ini dapat menambah wawasan pembaca dan dapat menjadi sumber referensi.

Melalui buku laporan ini penulis juga ingin menyampaikan rasa terima kasih kepada orang-orang yang telah membantu menyusun laporan kerja praktik baik secara langsung maupun tidak langsung antara lain:

1. Kedua orang tua penulis.
2. Bapak Dr. Yudhi Purwananto, S.Kom., M.Kom. selaku dosen pembimbing kerja praktik
3. Bapak Raden Muhammad Rizal Ghazali S.M., M.B.A. selaku pembimbing lapangan selama kerja praktik berlangsung.
4. Teman-teman penulis yang senantiasa memberikan semangat ketika penulis melaksanakan KP.

Surabaya, 26 September 2025
Helsa Sriprameswari Putri dan Aurelia Natasya Putri

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

1.1. Latar Belakang

Perkembangan teknologi informasi telah mendorong perusahaan untuk memanfaatkan sistem digital dalam meningkatkan kinerja dan produktivitas tenaga penjual. PT. Telkom Indonesia V memiliki beragam produk, seperti HSI, WMS, Netmonk, OCA, Pijar, dan Eazy, yang perlu dikelola dan dipantau penjualannya secara efektif. Sistem manajemen penjualan tradisional yang bersifat manual seringkali kurang efisien dan memerlukan waktu yang lama untuk memproses data serta memantau kinerja *sales agent*.

Kurangnya sistem yang terintegrasi juga dapat mengurangi motivasi *sales agent* karena tidak ada feedback atau pemantauan performa secara real-time. Hal ini membuat perusahaan kesulitan dalam mendorong pencapaian target penjualan yang optimal. Oleh karena itu, dibutuhkan sebuah aplikasi yang mampu mengelola data *sales agent*, memantau performa penjualan, dan memberikan informasi secara cepat dan akurat.

Gamifikasi menjadi salah satu pendekatan yang efektif untuk meningkatkan keterlibatan dan motivasi pengguna. Dengan mengaplikasikan elemen permainan seperti poin, badge, dan *leaderboard*, *sales agent* dapat melihat peringkat mereka berdasarkan penjualan produk Telkom, sehingga tercipta persaingan sehat dan dorongan untuk meningkatkan performa. *Leaderboard* ini menjadi sarana visual yang mempermudah manajemen dalam menilai kinerja masing-masing *sales agent*.

Berdasarkan kebutuhan tersebut, perancangan dan implementasi aplikasi web manajemen *sales agen*t

berbasis CRUD, gamifikasi, dan *leaderboard* diharapkan dapat membantu PT. Telkom Indonesia V dalam meningkatkan efisiensi pengelolaan data, memotivasi *sales agent*, dan mendukung strategi perusahaan dalam mencapai target penjualan produk secara lebih optimal.

1.2. Tujuan

Tujuan kerja praktik ini adalah menyelesaikan kewajiban nilai kerja praktik sebesar 4 sks dan membantu PT. Telkom Indonesia Regional V dalam menyelesaikan permasalahan monitoring dan manajemen *sales agent* melalui pengembangan aplikasi web.

1.3. Manfaat

Manfaat yang diperoleh dengan adanya aplikasi web manajemen *sales agent* ini antara lain adalah mempermudah perusahaan dalam melakukan monitoring dan manajemen data *sales agent* secara terpusat melalui sistem online. Selain itu, aplikasi ini dilengkapi dengan fitur *leaderboard* yang membantu menampilkan peringkat kinerja *sales agent* secara transparan, serta elemen gamifikasi yang dapat meningkatkan motivasi dan produktivitas *sales agent*. Dengan demikian, proses pengelolaan *sales agent* menjadi lebih efisien tanpa harus dilakukan secara manual.

1.4. Rumusan Masalah

Rumusan masalah dari kerja praktik ini adalah sebagai berikut:

1. Bagaimana merancang dan mengimplementasikan aplikasi web yang dapat mempermudah proses manajemen dan monitoring *sales agent* pada PT. Telkom Indonesia Regional V?
2. Bagaimana menerapkan gamifikasi dalam aplikasi web untuk meningkatkan motivasi dan produktivitas *sales agent*?

3. Bagaimana menampilkan peringkat kinerja *sales agent* secara transparan melalui fitur *leaderboard*?

1.5. Lokasi dan Waktu Kerja Praktik

Sehubungan dengan adanya kebijakan Work From Office (WFO), pengerjaan kerja praktik ini dilakukan secara offline di kantor setiap hari Senin hingga Jumat. Adapun kerja praktik dimulai pada tanggal 30 Juni 2025 hingga 30 September 2025.

1.6. Metodologi Kerja Praktik

Metodologi dalam pembuatan buku kerja praktik meliputi :

1.6.1. Perumusan Masalah

Untuk mengetahui kebutuhan dari aplikasi web, saya mengikuti rapat bersama tim developer PT. Telkom Indonesia Regional V serta didampingi oleh mentor, Bapak Rizal. Dalam rapat tersebut dijelaskan mengenai konsep dan alur kerja sistem manajemen *sales agent* yang akan dikembangkan. Tim developer kemudian merumuskan fitur-fitur utama yang perlu diterapkan, antara lain pengelolaan data *sales agent* (CRUD), *leaderboard* untuk menampilkan peringkat kinerja, serta elemen gamifikasi untuk meningkatkan motivasi. Selain itu, dibahas pula estimasi jumlah pengguna aplikasi sehingga dapat ditentukan arsitektur sistem yang sesuai dan mampu mendukung kebutuhan perusahaan.

1.6.2. Studi Literatur

Setelah mendapat gambaran bagaimana sistem tersebut akan berjalan, kami diberitahu tinjauan teknologi yang digunakan untuk membangun aplikasi web. Teknologi yang dipakai meliputi Laravel sebagai framework backend, PostgreSQL sebagai basis data, Tailwind CSS dan template Konrix untuk antarmuka

pengguna, serta Vite untuk optimasi build aset. Selain itu, Git digunakan sebagai version control agar memudahkan kolaborasi. Kami juga dijelaskan mengenai aturan dalam penulisan konfigurasi agar dapat dipahami oleh pengembang lain.

1.6.3. Analisis dan Perancangan Sistem

Setelah tinjauan yang dipakai telah diberitahu, untuk merancang sistem yang baik perlu adanya sebuah desain arsitektur sistem. Pada website ini tim developer setuju menggunakan arsitektur desain MVC (Model - View - Controller).

1.6.4. Implementasi Sistem

Implementasi merupakan tahap realisasi dari hasil perancangan sistem. Pada tahap ini, kami melakukan pembangunan aplikasi secara full stack, mulai dari perancangan basis data, pengembangan backend, hingga pembuatan antarmuka pengguna, serta integrasi seluruh komponen agar aplikasi dapat berjalan sesuai kebutuhan.

1.6.5. Pengujian dan Evaluasi

Setelah website yang telah direncanakan telah jadi, perlu adanya evaluasi untuk menguji apakah website sesuai dengan harapan client. Jika masih belum sesuai atau perlu menambah fitur, rapat akan dilakukan lagi untuk menentukan fitur - fitur apa saja yang perlu diperbaiki atau ditambah.

1.6.6. Kesimpulan dan Saran

Pengujian yang dilakukan ini telah memenuhi syarat yang diinginkan, dan berjalan dengan baik dan lancar.

1.7. Sistematika Laporan

1.7.1. Bab I Pendahuluan

Bab ini berisi latar belakang, tujuan, manfaat, rumusan masalah, lokasi dan waktu kerja praktik, metodologi, dan sistematika laporan.

1.7.2. Bab II Profil Perusahaan

Bab ini berisi gambaran umum PT Telkom Indonesia Regional V mulai dari profil, lokasi perusahaan.

1.7.3. Bab III Tinjauan Pustaka

Bab ini berisi dasar teori dari teknologi yang digunakan dalam menyelesaikan proyek kerja praktik.

1.7.4. Bab IV Analisis dan Perancangan Infrastruktur Sistem

Bab ini berisi mengenai tahap analisis sistem aplikasi dalam menyelesaikan proyek kerja praktik.

1.7.5. Bab V Implementasi Sistem

Bab ini berisi uraian tahap - tahap yang dilakukan untuk proses implementasi aplikasi.

1.7.6. Bab VI Pengujian dan Evaluasi

Bab ini berisi hasil uji coba dan evaluasi dari aplikasi yang telah dikembangkan selama pelaksanaan kerja praktik.

1.7.7. Bab VII Kesimpulan dan Saran

Bab ini berisi kesimpulan dan saran yang didapat dari proses pelaksanaan kerja praktik.

[Halaman ini sengaja dikosongkan]

BAB II
PROFIL PERUSAHAAN

2.1. Profil PT Telkom Indonesia Regional V

PT Telekomunikasi Indonesia (Persero) Tbk, atau lebih dikenal dengan Telkom Indonesia, merupakan perusahaan penyedia layanan dan jaringan telekomunikasi terbesar di Indonesia. Telkom menyediakan beragam layanan di bidang Information and Communication (Infocomm), seperti layanan telepon tetap kabel (*fixed wireline*), telepon tetap nirkabel (*fixed wireless*), layanan seluler (*mobile service*), data dan internet, multimedia, serta jasa interkoneksi, baik secara langsung maupun melalui anak perusahaannya.

Sejarah Telkom bermula dari badan usaha swasta penyedia layanan pos dan telegraf bernama Jawatan. Pada tahun 1961 statusnya diubah menjadi Perusahaan Negara Pos dan Telekomunikasi (PN Postel). Kemudian PN Postel dipecah menjadi Perusahaan Negara Pos dan Giro (PN Pos & Giro) dan Perusahaan Negara Telekomunikasi (PN Telekomunikasi). Pada tahun 1974, PN Telekomunikasi berubah menjadi Perusahaan Umum Telekomunikasi (Perumtel) yang bertugas menyelenggarakan jasa telekomunikasi nasional maupun internasional. Pada tanggal 14 November 1995, resmi digunakan nama PT Telekomunikasi Indonesia (Persero) Tbk., yang hingga kini menjadi perusahaan telekomunikasi terbesar di Indonesia.

Secara organisasi, Telkom memiliki beberapa regional operasional yang membawahi wilayah-wilayah di Indonesia. Salah satunya adalah Telkom Regional V Jawa Timur (Telkom Regional V Jatim) yang berkedudukan di Surabaya. Regional V bertanggung jawab atas penyediaan,

pengelolaan, dan pengembangan layanan telekomunikasi di seluruh wilayah Jawa Timur. Unit ini berperan dalam mendukung strategi Telkom untuk memperluas penetrasi layanan digital, meningkatkan kualitas jaringan, serta memastikan layanan kepada pelanggan di Jawa Timur berjalan optimal.

Dengan keberadaan Telkom Regional V Jatim, perusahaan dapat lebih fokus dalam memberikan pelayanan dan solusi digital yang sesuai dengan kebutuhan masyarakat dan instansi di wilayah Jawa Timur, termasuk dalam pengembangan aplikasi internal, peningkatan infrastruktur jaringan, serta penyediaan layanan digital inovatif.

2.2. Profil Unit RSMES (Regional SME, Enterprise, and Government Service)

RSMES (Regional SME, Enterprise, and Government Service) adalah unit regional PT Telkom Indonesia yang fokus pada pengembangan dan pelayanan segmen usaha kecil dan menengah (UKM), korporasi, serta instansi pemerintah di wilayah Jawa Timur, Bali dan Nusa Tenggara. Unit ini menjadi bagian penting dari strategi Telkom untuk memperkuat bisnis regional dan mendukung transformasi digital dengan menyediakan solusi digital komprehensif melalui platform Indibiz, yang mencakup layanan konektivitas enterprise, pusat data, cloud, dan layanan digital lainnya.

Selain itu, RSMES aktif mendukung program tanggung jawab sosial dan lingkungan (TJSL) Telkom, termasuk inisiatif pemberdayaan masyarakat dan inovasi berkelanjutan. Dengan pendekatan yang holistik dan berbasis teknologi, RSMES berperan penting dalam memperkuat ekosistem digital, meningkatkan daya saing

bisnis lokal, dan mendukung pembangunan ekonomi digital yang inklusif di wilayahnya.

2.3. Lokasi PT Telkom Indonesia Regional V

Jl. Dr. Ir. H. Soekarno No.175, Klampis Ngasem, Kec. Sukolilo, Surabaya, Jawa Timur 60116

2.4. Visi dan Misi Perusahaan

Visi PT. Telkom Indonesia Regional V Jatim adalah:

"Menjadi perusahaan yang unggul dalam penyelenggaraan TIMES di kawasan regional."

Untuk mendukung pencapaian visi tersebut, PT. Telkom Indonesia Regional V Jatim memiliki misi sebagai berikut:

1. Menyediakan layanan TIMES (Telecommunication, Information, Media, Edutainment, and Services) yang berkualitas tinggi dengan harga yang kompetitif.
2. Menjadi model pengelolaan korporasi terbaik di Indonesia.

[Halaman ini sengaja dikosongkan]

BAB III
TINJAUAN PUSTAKA

3.1. PHP

PHP adalah bahasa pemrograman server-side yang dirancang untuk pengembangan web dinamis dan dapat di-embed ke dalam HTML. PHP memungkinkan pembuatan halaman web interaktif, pengelolaan form, koneksi ke basis data, serta pengolahan data secara real-time. Dengan sintaks yang mudah dipahami dan dukungan komunitas besar, PHP banyak digunakan dalam pengembangan aplikasi web modern, termasuk framework seperti Laravel yang mempermudah pengelolaan logika backend.

3.2. Laravel

Laravel adalah framework PHP berbasis arsitektur MVC (Model-View-Controller) yang dirancang untuk mempermudah pengembangan aplikasi web skala kecil hingga besar. Laravel menyediakan fitur bawaan seperti routing, middleware, ORM (Eloquent), autentikasi, dan template engine Blade, sehingga pengembang dapat membuat aplikasi web dengan struktur kode yang terorganisir, aman, dan mudah dipelihara. Framework ini juga mendukung integrasi dengan database, sistem caching, dan task scheduling, sehingga cocok digunakan dalam pengembangan aplikasi fullstack modern.

3.3. HTML

HTML (HyperText Markup Language) adalah bahasa markup standar yang digunakan untuk membuat dan menyusun halaman web. HTML berfungsi untuk mendefinisikan struktur konten, seperti teks, gambar, tautan, tabel, dan form, sehingga browser dapat menampilkan halaman web dengan benar. HTML juga menjadi dasar bagi teknologi web lainnya, seperti CSS untuk styling dan JavaScript untuk interaktivitas, sehingga

menjadi komponen utama dalam pengembangan aplikasi web modern.

3.4. Tailwind CSS

Tailwind CSS adalah framework CSS utility-first yang memungkinkan pengembang membangun antarmuka web secara cepat dan konsisten dengan menggunakan kelas-kelas kecil yang siap pakai. Berbeda dengan framework CSS tradisional, Tailwind tidak menyediakan komponen siap pakai, melainkan menyediakan utilitas untuk styling, layout, dan responsivitas yang dapat dikombinasikan sesuai kebutuhan. Pendekatan ini mempermudah pembuatan desain yang fleksibel, terstruktur, dan mudah dipelihara, terutama dalam pengembangan aplikasi web modern.

3.5. JavaScript

JavaScript adalah bahasa pemrograman sisi klien (client-side) yang digunakan untuk membuat halaman web menjadi interaktif. Dengan JavaScript, pengembang dapat memanipulasi elemen HTML dan CSS secara dinamis, melakukan validasi form, menangani event pengguna, serta berkomunikasi dengan server menggunakan teknologi seperti AJAX atau Fetch API. JavaScript menjadi salah satu komponen utama dalam pengembangan aplikasi web modern, sering dipadukan dengan framework atau library seperti React, Vue, maupun digunakan langsung bersama HTML dan CSS.

3.6. PostgreSQL

PostgreSQL adalah sistem manajemen basis data relasional (RDBMS) open-source yang mendukung penyimpanan, pengelolaan, dan pengambilan data secara efisien. PostgreSQL mendukung fitur-fitur canggih seperti transaksi ACID, indexing, foreign key, view, dan prosedur tersimpan, sehingga sangat cocok digunakan untuk aplikasi web yang membutuhkan konsistensi dan integritas data tinggi. Dengan kemampuan skalabilitas dan

kompatibilitasnya yang baik, PostgreSQL banyak dipilih dalam pengembangan aplikasi modern, termasuk aplikasi fullstack berbasis Laravel.

3.7. Vite

Vite adalah build tool modern untuk pengembangan web yang menyediakan server development cepat dan proses build produksi yang efisien. Vite menggunakan modul ES (ES Modules) untuk menyajikan file secara langsung ke browser selama pengembangan, sehingga mempercepat hot module replacement (HMR) dan meningkatkan produktivitas pengembang. Selain itu, Vite dapat diintegrasikan dengan berbagai framework frontend dan backend, termasuk Laravel, untuk mengoptimalkan bundling aset CSS dan JavaScript..

[Halaman ini sengaja dikosongkan]

BAB IV
ANALISIS DAN PERANCANGAN INFRASTRUKTUR SISTEM

4.1. Analisis Sistem

Pada bab ini akan dijelaskan mengenai proses perancangan dan implementasi aplikasi web manajemen *sales agent* dengan gamifikasi dan *leaderboard* di PT. Telkom Indonesia V. Penjelasan ini mencakup analisis dan perancangan sistem yang akan dibangun, serta kebutuhan fungsional dan non-fungsional aplikasi. Hal tersebut dijelaskan ke dalam dua bagian, definisi umum aplikasi dan analisis kebutuhan fungsional

4.1.1. Definisi Umum Aplikasi

Aplikasi web manajemen *sales agent* dengan gamifikasi dan *leaderboard* adalah sistem berbasis web untuk mengelola data penjualan produk Telkom, memantau kinerja *sales agent,* dan menampilkan peringkat penjualan. Sistem memiliki dua tampilan, yaitu *sales agent* untuk input penjualan dan melihat *leaderboard*, serta admin untuk mengelola data dan memantau performa secara keseluruhan.

4.1.2. Analisis Kebutuhan

Dalam aplikasi ini, terdapat fungsi-fungsi yang harus dipenuhi oleh sistem. Kebutuhan ini terbagi ke dalam dua jenis, yakni kebutuhan fungsional dan kebutuhan non- fungsional.

a. Kebutuhan Fungsional

Kebutuhan fungsional pada aplikasi ini menjelaskan bagaimana sistem bekerja. Kebutuhan fungsional dari prototipe antarmuka aplikasi web manajemen *sales agent* dengan

gamifikasi dan *leaderboard* berbasis PHP dengan Laravel dijelaskan pada Tabel 4.1.

Tabel 4.1 Kebutuhan Fungsional Aplikasi Website Sales Agent

Kode Kebutuhan	Deskripsi Kebutuhan
F-001	Melihat data seluruh *sales agent*.
F-002	Menambahkan *sales agent* baru.
F-003	Mengedit data *sales agent*.
F-004	Menghapus data *sales agent*.
F-005	Melakukan filter *sales agent* berdasarkan jenis, status, witel, telda, agency, nama, dan kode.
F-006	Melihat data order.
F-007	Melakukan filter order berdasarkan witel, telda sales, dan order id.
F-008	Mengedit kepemilikan order *sales agent*.
F-009	Melihat *leaderboard* penjualan sales sesuai rentang tanggal tertentu.

F-0010	Melakukan filter *leaderboard* berdasarkan jenis, status, witel, telda, agency, nama, dan kode sales.
F-0011	Mengurutkan *leaderboard* berdasarkan tanggal, nama *sales agent*, agency, total penjualan, ranking, serta jenis produk (HSI,WMS, Netmonk, OCA, Pijar, Eazy).
F-0012	Melihat detail order penjualan produk di dalam *leaderboard*.
F-0013	Melihat performa *sales agent* yang mencakup skor total, ranking, benefit, target, realisasi, pencapaian, pertumbuhan, dan skor tiap produk digital dalam gamifikasi.
F-0014	Melakukan filter gamifikasi berdasarkan bulan, witel, telda, agency, jenis, kode, dan nama *sales agent*.
F-0015	Mengurutkan gamifikasi berdasarkan nama, performa, target, realisasi,

	pencapaian, pertumbuhan, skor, skor total, dan ranking.
F-0016	Melihat detail order penjualan untuk tiap produk di dalam gamifikasi.
F-0017	Menambahkan target penjualan *sales agent* untuk tiap produk.
F-0018	Menetapkan bobot kriteria tiap produk.

Tabel 4.1 menunjukkan kebutuhan fungsional aplikasi web berbasis Laravel. Aplikasi ini dapat diakses oleh aktor sesuai dengan perannya (role), dan setiap aktor memiliki fitur yang berbeda berdasarkan peran tersebut.

b. Kebutuhan Non-Fungsional

Kebutuhan non-fungsional adalah kebutuhan pengguna untuk mendefinisikan bagaimana batasan dan karakteristik dari sebuah sistem yang dibangun. Kebutuhan non- fungsional dari aplikasi web *sales agent* terdapat pada Tabel 4.2.

Tabel 4.2 Kebutuhan Non-Fungsional Aplikasi Website Sales Agent

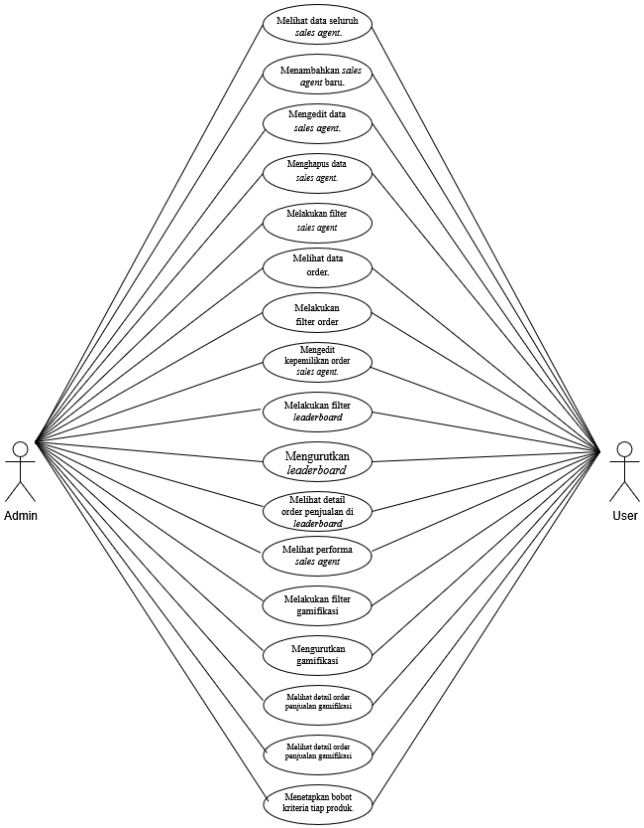
Kode Kebutuhan	Deskripsi Kebutuhan
F-001	Sistem dapat diakses oleh pengguna melalui browser modern (Chrome, Firefox, Edge, dan Safari).
F-002	Sistem dapat diakses melalui perangkat desktop maupun mobile (responsif).
F-003	Sistem memiliki tampilan antarmuka yang sederhana, konsisten, dan mudah dipahami oleh pengguna.
F-004	Sistem memiliki audit log untuk merekam setiap aktivitas yang dilakukan oleh admin maupun *user*.
F-005	Sistem mampu menangani banyak permintaan (request per detik) untuk memastikan respon

	cepat pada saat mengakses data *sales agent*, data order, *leaderboard*, dan gamifikasi.
F-006	Sistem memastikan bahwa seluruh data dalam aplikasi terlindung dari akses yang tidak berwenang melalui mekanisme otentikasi, otorisasi, dan enkripsi.

Pada Tabel 4.2 terdapat kebutuhan non-fungsional prototipe antarmuka aplikasi website *sales agent* berbasis laravel yang mencakup aspek aksesibilitas pengguna, kompatibilitas dengan sistem operasi, tampilan antarmuka yang sederhana dan mudah dipahami, serta perlindungan keamanan data pengguna.

4.2. Diagram Kasus Penggunaan

Pembahasan dengan pembimbing lapangan tentang fitur-fitur yang perlu ada dalam aplikasi website s*ales agent* berbasis Laravel menghasilkan beberapa fitur yang dijadikan diagram kasus penggunaan (*Use Case Diagram*) sehingga memudahkan untuk dipahami. *Use Case Diagram* yang telah dibuat dapat dilihat pada Gambar 4.2.1.



Gambar 4.2.1 Use Case Diagram Aplikasi Website Sales Agent

Pada Gambar 4.2.1, ditunjukkan diagram *use case* dari aplikasi web Sales Agent berbasis gamifikasi dan leaderboard. Diagram ini menggambarkan dua aktor utama, yaitu Admin dan *User*: Admin dapat melakukan seluruh aktivitas manajemen data dalam sistem, meliputi

pengelolaan data sales agent, order, leaderboard, target, serta bobot produk. Admin juga memiliki hak akses untuk melihat dan memperbarui data performa sales agent. *User* hanya dapat mengakses fitur yang bersifat informatif, seperti melihat leaderboard, performa pribadi, dan rekomendasi sistem, namun tidak memiliki wewenang untuk mengedit data. Use case ini memperlihatkan perbedaan hak akses antara kedua aktor, yang mencerminkan struktur kontrol dan pembagian tanggung jawab dalam sistem.

4.3 Spesifikasi Kasus Penggunaan

4.3.1 Melihat Data Seluruh Sales Agent.

Tabel 4.3 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk melihat data seluruh *sales agent*.

Tabel 4.3 Use Case Aplikasi Website *Sales Agent* untuk Melihat Data Seluruh *Sales Agent*

Nama	Melihat data seluruh *sales agent*.
Kode	UC-001
Deskripsi	Use case ini menggambarkan proses pengguna (admin/*user*) dalam melihat daftar seluruh *sales agent* yang tersimpan pada sistem.
Tipe	Fungsional

Pemicu	Pengguna memilih menu "*Sales Agent*" pada antarmuka aplikasi.
Aktor	Admin, *User*
Kondisi Awal	Pengguna telah berhasil login ke dalam sistem untuk mengakses dashboard.
Kondisi Akhir	Sistem menampilkan daftar data seluruh *sales agent* sesuai hak akses pengguna.
Alur Kejadian Secara Normal	1. Pengguna login ke aplikasi. 2. Pengguna memilih menu "Data *Sales Agent*". 3. Sistem memproses permintaan dan mengambil data dari basis data. 4. Sistem menampilkan daftar seluruh *sales agent* kepada pengguna.
Alur Kejadian Alternatif	Jika data terlalu banyak, sistem menyediakan fitur pencarian dan filter agar pengguna dapat menemukan data *sales agent* dengan lebih cepat.
Pengecualian	- Jika basis data tidak dapat diakses, sistem menampilkan pesan error.

	- Jika pengguna tidak memiliki hak akses, sistem menolak permintaan.

4.3.2 Menambahkan *Sales Agent* Baru.

Tabel 4.4 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk menambahkan *sales agent* baru.

Tabel 4.4 Use Case Aplikasi Website *Sales Agent* untuk Menambahkan *Sales Agent* Baru

Nama	Menambahkan *Sales Agent* Baru
Kode	UC-002
Deskripsi	Use case ini menggambarkan proses admin atau *user* dalam menambahkan data *sales agent* baru ke dalam sistem.
Tipe	Fungsional
Pemicu	Admin menekan *button* "Tambah *Sales Agent*" pada antarmuka aplikasi.
Aktor	Admin, *User*
Kondisi Awal	*Button* "Tambah *Sales Agent*" ditampilkan di menu *sales agent*.

Kondisi Akhir	Data *sales agent* baru berhasil tersimpan ke dalam basis data dan muncul di daftar *sales agent*.
Alur Kejadian Secara Normal	1. Admin memilih tombol "Tambah *Sales Agent*". 2. Sistem menampilkan form penambahan *sales agent* baru. 3. Admin mengisi data *sales agent* baru. 4. Admin menekan tombol "Simpan". 5. Sistem memvalidasi dan menyimpan data.
Alur Kejadian Alternatif	- Jika data tidak lengkap, sistem menampilkan pesan untuk melengkapi data. - Admin dapat membatalkan proses penambahan.
Pengecualian	- Jika terjadi kesalahan koneksi ke basis data, sistem menampilkan pesan error. - Jika ID *sales agent* sudah ada, sistem menolak penyimpanan dan menampilkan pesan peringatan.

4.3.3 Mengedit Data *Sales Agent*

Tabel 4.5 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk melihat data seluruh *sales agent*.

Tabel 4.5 Use Case Aplikasi Website *Sales Agent* untuk Mengedit Data *Sales Agent*

Nama	Mengedit Data *Sales Agent*
Kode	UC-003
Deskripsi	Use case ini menggambarkan proses admin dalam memperbarui atau mengubah data *sales agent* yang sudah ada di sistem.
Tipe	Fungsional
Pemicu	Admin menekan tombol "Edit" pada data *sales agent* yang ingin diperbarui.
Aktor	Admin atau *User*
Kondisi Awal	Admin telah login ke dalam sistem dan berada pada halaman daftar data *sales agent*.
Kondisi Akhir	Data *sales agent* yang dipilih berhasil diperbarui dan tersimpan ke dalam basis data.

Alur Kejadian Secara Normal	1. Admin login ke aplikasi. 2. Admin membuka halaman daftar *sales agent*. 3. Admin memilih tombol "Edit" pada *sales agent* tertentu. 4. Sistem menampilkan form edit dengan data *sales agent* yang sudah ada. 5. Admin melakukan perubahan pada data. 6. Admin menekan tombol "Simpan". 7. Sistem memvalidasi data yang diperbarui. 8. Jika valid, sistem menyimpan perubahan ke basis data.
Alur Kejadian Alternatif	- Jika admin tidak jadi mengubah data, admin dapat menekan tombol "Batal" dan sistem kembali ke daftar *sales agent* tanpa perubahan.
Pengecualian	- Jika data yang dimasukkan tidak valid, sistem menampilkan pesan error.

4.3.4 Menghapus Data *Sales Agent*.

Tabel 4.6 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk menghapus data *sales agent*.

Tabel 4.6 Use Case Aplikasi Website *Sales Agent* untuk Menghapus Data *Sales Agent*.

Nama	Menghapus Data *Sales Agent*
Kode	UC-004
Deskripsi	Use case ini menggambarkan proses admin dalam menghapus data *sales agent* yang sudah tidak diperlukan dari sistem.
Tipe	Fungsional
Pemicu	Admin menekan tombol "Hapus" pada data *sales agent* tertentu di daftar *sales agent*.
Aktor	Admin atau *User*
Kondisi Awal	Admin telah login ke sistem dan berada pada halaman daftar data *sales agent*.
Kondisi Akhir	Data *sales agent* yang dipilih berhasil dihapus dari basis data dan tidak lagi ditampilkan pada daftar *sales agent*.

Alur Kejadian Secara Normal	1. Admin login ke aplikasi. 2. Admin membuka halaman daftar *sales agent*. 3. Admin memilih tombol "Hapus" pada *sales agent* tertentu. 4. Sistem menampilkan dialog konfirmasi penghapusan. 5. Admin menekan tombol "Ya" untuk mengonfirmasi. 6. Sistem menghapus data *sales agent* dari basis data.
Alur Kejadian Alternatif	Jika admin menekan tombol "Batal" pada dialog konfirmasi, maka sistem tidak menghapus data dan kembali ke daftar *sales agent*.
Pengecualian	- Jika koneksi ke basis data gagal, sistem menampilkan pesan error. - Jika data *sales agent* yang dipilih sudah tidak ada, sistem menampilkan pesan peringatan.

4.3.5 Melakukan Filter *Sales Agent* Berdasarkan Jenis, Status, Witel, Telda, Agency, Nama, dan Kode.

Tabel 4.7 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk melakukan filter *sales agent* berdasarkan jenis, status, witel, telda, agency, nama, dan kode.

Tabel 4.7 Use Case Aplikasi Website *Sales Agent* untuk Melakukan Filter *Sales Agent* Berdasarkan Jenis, Status, Witel, Telda, Agency, Nama, dan Kode.

Nama	Melakukan Filter *Sales Agent* Berdasarkan Jenis, Status, Witel, Telda, Agency, Nama, dan Kode.
Kode	UC-005
Deskripsi	Use case ini menggambarkan proses pengguna dalam melakukan filter data *sales agent* berdasarkan kriteria tertentu (Jenis, Status, Witel, Telda, Agency, Nama, dan Kode Sales).
Tipe	Fungsional
Pemicu	Pengguna memilih opsi filter dan menetapkan parameter pencarian pada halaman daftar *sales agent*.
Aktor	Admin atau *User*

Kondisi Awal	Pengguna telah login ke dalam sistem dan berada pada halaman daftar *sales agent*.
Kondisi Akhir	Sistem menampilkan daftar *sales agent* sesuai dengan filter yang dipilih oleh pengguna.
Alur Kejadian Secara Normal	1. Pengguna login ke aplikasi. 2. Pengguna membuka halaman daftar *sales agent*. 3. Pengguna memilih kriteria filter (Jenis, Status, Witel, Telda, Agency, Nama, atau Kode). 4. Pengguna mengisi parameter filter sesuai kebutuhan. 5. Pengguna menekan tombol "Terapkan Filter". 6. Sistem memproses permintaan dan menampilkan data sesuai kriteria filter.
Alur Kejadian Alternatif	- Jika filter tidak sesuai, sistem menampilkan pesan "Data tidak ditemukan". - Pengguna dapat menghapus filter untuk kembali ke daftar lengkap.

Pengecualian	Jika parameter filter tidak valid, sistem menampilkan pesan kesalahan.

4.3.6 Melihat Data Order.

Tabel 4.8 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk melihat data order.

Tabel 4.8 Use Case Aplikasi Website *Sales Agent* untuk Melihat Data Order.

Nama	Melihat Data Order
Kode	UC-006
Deskripsi	Use case ini menggambarkan proses pengguna dalam melihat daftar data order yang tersimpan di dalam sistem.
Tipe	Fungsional
Pemicu	Pengguna memilih menu “Order" pada antarmuka aplikasi.
Aktor	Admin atau User
Kondisi Awal	Pengguna telah berhasil login dan mengakses dashboard sistem.

Kondisi Akhir	Sistem menampilkan daftar data order.
Alur Kejadian Secara Normal	1. Pengguna login ke aplikasi. 2. Pengguna membuka menu "Data Order". 3. Sistem mengambil data order dari basis data. 4. Sistem menampilkan daftar order kepada pengguna.
Alur Kejadian Alternatif	- Jika data order sangat banyak, sistem menyediakan fitur pencarian dan filter agar pengguna lebih mudah menemukan data.
Pengecualian	- Jika koneksi ke basis data gagal, sistem menampilkan pesan error. - Jika pengguna tidak memiliki hak akses, sistem menolak permintaan.

4.3.7 Melakukan Filter Order Berdasarkan Witel, Telda Sales, dan Order ID.

Tabel 4.9 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk melakukan filter order Berdasarkan Witel, Telda Sales, dan Order ID.

Tabel 4.9 Use Case Aplikasi Website *Sales Agent* untuk Melakukan Filter Order Berdasarkan Witel, Telda Sales, dan Order ID.

Nama	Melakukan Filter Order Berdasarkan Witel, Telda Sales, dan Order ID.
Kode	UC-007
Deskripsi	Use case ini menggambarkan proses pengguna dalam memfilter data order berdasarkan kriteria tertentu seperti Witel, Telda Sales, dan Order ID.
Tipe	Fungsional
Pemicu	Pengguna memilih opsi filter dan menetapkan parameter pencarian pada halaman daftar order.
Aktor	Admin atau *User*
Kondisi Awal	Pengguna telah login ke dalam sistem dan berada pada halaman daftar order.
Kondisi Akhir	Sistem menampilkan daftar order sesuai dengan filter yang dipilih pengguna.

Alur Kejadian Secara Normal	1. Pengguna login ke aplikasi. 2. Pengguna membuka halaman daftar order. 3. Pengguna memilih kriteria filter (Witel, Telda Sales, atau Order ID). 4. Pengguna mengisi parameter filter. 5. Pengguna menekan tombol "Terapkan Filter". 6. Sistem memproses filter dan menampilkan daftar order sesuai kriteria.
Alur Kejadian Alternatif	- Pengguna dapat menghapus filter untuk menampilkan semua data order kembali.
Pengecualian	- Jika koneksi ke basis data gagal, sistem menampilkan pesan error. - Jika parameter filter tidak valid, sistem menampilkan pesan kesalahan.

4.3.8 Mengedit Kepemilikan Order *Sales Agent*.

Tabel 4.10 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk mengedit kepemilikan order *sales agent*.

Tabel 4.10 Use Case Aplikasi Website *Sales Agent* untuk Mengedit Kepemilikan Order *Sales Agent*.

Nama	Mengedit Kepemilikan Order *Sales Agent*
Kode	UC-008
Deskripsi	Use case ini menggambarkan proses admin dalam memperbarui atau memindahkan kepemilikan order dari satu *sales agent* ke *sales agent* lain.
Tipe	Fungsional
Pemicu	Admin menekan tombol "Edit Order" pada order tertentu.
Aktor	Admin atau *User*
Kondisi Awal	Admin telah login ke sistem dan berada pada halaman daftar order.
Kondisi Akhir	Kepemilikan order berhasil diperbarui dan tersimpan di basis data, serta sistem menampilkan informasi terbaru pada daftar order.
Alur Kejadian Secara Normal	1. Admin login ke aplikasi. 2. Admin membuka halaman

	daftar order. 3. Admin memilih tombol "Edit" pada order yang ingin diperbarui. 4. Sistem menampilkan form edit dengan data order yang ada. 5. Admin memilih *sales agent* baru untuk kepemilikan order. 6. Admin menekan tombol "Simpan". 7. Sistem memvalidasi perubahan dan menyimpan ke basis data.
Alur Kejadian Alternatif	- Jika admin membatalkan proses edit, perubahan tidak disimpan dan sistem kembali ke daftar order.
Pengecualian	- Jika koneksi ke basis data gagal, sistem menampilkan pesan error - Jika data *sales agent* baru tidak valid, sistem menampilkan pesan kesalahan.

4.3.9 Melihat *Leaderboard* Penjualan Sales Sesuai Rentang Tanggal Tertentu.

Tabel 4.11 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk melihat *leaderboard* penjualan sales sesuai rentang tanggal tertentu.

Tabel 4.11 Use Case Aplikasi Website *Sales Agent* untuk Melihat *Leaderboard* Penjualan Sales Sesuai Rentang Tanggal Tertentu.

Nama	Melihat *Leaderboard* Penjualan Sales Sesuai Rentang Tanggal Tertentu.
Kode	UC-009
Deskripsi	Use case ini memungkinkan admin untuk melihat peringkat *sales agent* berdasarkan jumlah penjualan produk Telkom tertentu (HSI, WMS, Netmonk, OCA, Pijar, Eazy) dalam rentang tanggal yang dipilih.
Tipe	Fungsional
Pemicu	Admin menekan tombol "*Leaderboard*" pada sidebar
Aktor	Admin atau *User*
Kondisi Awal	Pengguna telah login ke

	sistem dan berada di dashboard penjualan.
Kondisi Akhir	Pengguna berhasil melihat *leaderboard sales agent* beserta total penjualan produk Telkom yang dijual dalam periode tersebut.
Alur Kejadian Secara Normal	1. Pengguna memilih menu "*Leaderboard* Penjualan Produk Telkom". 2. Sistem menampilkan filter untuk memilih rentang tanggal dan jenis produk Telkom. 3. Pengguna memasukkan tanggal mulai dan tanggal akhir, serta memilih produk (atau semua produk). 4. Pengguna menekan tombol "Filter". 5. Sistem menampilkan daftar *sales agent* dengan total penjualan dan peringkat sesuai produk dan rentang tanggal.
Alur Kejadian Alternatif	- Pengguna memilih rentang tanggal yang kosong atau tidak valid sehingga hasil kosong.
Pengecualian	- Jika tidak ada penjualan dalam rentang tanggal atau

	produk yang dipilih, sistem menampilkan pesan "Tidak ada penjualan untuk periode dan produk yang dipilih".

4.3.10 Melakukan Filter *Leaderboard* Berdasarkan Jenis, Status, Witel, Telda, Agency, Nama, dan Kode Sales

Tabel 4.12 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk melakukan filter *leaderboard* berdasarkan jenis, status, witel, telda, supreme, nama, dan kode sales.

Tabel 4.12 Use Case Aplikasi Website *Sales Agent* untuk Melakukan Filter *Leaderboard* Berdasarkan Jenis, Status, Witel, Telda, Agency, Nama, dan Kode Sales

Nama	Melakukan Filter *Leaderboard* Berdasarkan Jenis, Status, Witel, Telda, Agency, Nama, dan Kode Sales
Kode	UC-0010
Deskripsi	Use case ini menggambarkan proses admin dalam memfilter *leaderboard* penjualan sales berdasarkan kriteria seperti jenis produk, status, witel, telda, agency, nama sales, dan kode sales.

Tipe	Fungsional
Pemicu	Admin menekan tombol "Filter" pada halaman *leaderboard*.
Aktor	Admin atau *User*
Kondisi Awal	Pengguna telah login ke sistem dan berada pada halaman *leaderboard* penjualan sales.
Kondisi Akhir	*Leaderboard* berhasil menampilkan data sales yang sesuai dengan kriteria filter yang dipilih.
Alur Kejadian Secara Normal	1. Pengguna membuka halaman "*Leaderboard* Penjualan". 2. Sistem menampilkan opsi filter: Jenis Produk, Status, Witel, Telda, Agency, Nama, dan Kode Sales. 3. Pengguna memilih kriteria filter yang diinginkan. 4. Pengguna menekan tombol " Filter". 5. Sistem memproses filter dan menampilkan *leaderboard* yang sesuai kriteria.
Alur Kejadian	Jika pengguna tidak memilih

Alternatif	filter apapun, sistem menampilkan *leaderboard* lengkap tanpa filter.
Pengecualian	Jika koneksi ke basis data gagal, sistem menampilkan pesan error.

4.3.11 Mengurutkan *Leaderboard* Berdasarkan Tanggal, Nama *Sales Agent*, Agency, Total Penjualan, Ranking, serta Jenis Produk (HSI, WMS, Netmonk, OCA, Pijar, Eazy).

Tabel 4.13 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk mengurutkan *leaderboard* berdasarkan tanggal, nama *sales agent*, agency, total penjualan, ranking, serta jenis produk (HSI, WMS, Netmonk, OCA, Pijar, Eazy).

Tabel 4.13 Use Case Aplikasi Website *Sales Agent* untuk Mengurutkan *Leaderboard* Berdasarkan Tanggal, Nama *Sales Agent*, Agency, Total Penjualan, Ranking, serta Jenis Produk (HSI, WMS, Netmonk, OCA, Pijar, Eazy)

Nama	Mengurutkan *Leaderboard* Berdasarkan Tanggal, Nama *Sales Agent*, Agency, Total Penjualan, Ranking, serta Jenis Produk (HSI, WMS, Netmonk, OCA, Pijar, Eazy).
Kode	UC-011
Deskripsi	Use case ini menggambarkan

	proses admin atau *user* dalam mengurutkan *leaderboard* berdasarkan Tanggal, Nama *Sales Agent*, Agency, Total Penjualan, Ranking, serta Jenis Produk (HSI, WMS, Netmonk, OCA, Pijar, Eazy).
Tipe	Fungsi
Pemicu	*User* memilih tombol pengurutan (*sorting*) pada halaman *leaderboard*.
Aktor	Admin atau *User*
Kondisi Awal	- *User* sudah login ke sistem. - Halaman data *leaderboard* tersedia dalam sistem.
Kondisi Akhir	Sistem menampilkan *leaderboard* sesuai urutan yang dipilih *user*.
Alur Kejadian Secara Normal	1. *User* masuk ke halaman *leaderboard*. 2. Sistem menampilkan data *leaderboard* default (urutan standar). 3. *User* memilih kriteria pengurutan (Tanggal, Nama *Sales Agent*, Agency, Total Penjualan, Ranking, atau Jenis Produk). 4. Sistem memproses

	permintaan pengurutan. 5. Sistem menampilkan *leaderboard* sesuai kriteria urutan yang dipilih.
Alur Kejadian Alternatif	- Jika *user* memilih lebih dari satu kriteria pengurutan, sistem mengurutkan berdasarkan pengurutan terakhir. - *User* dapat mengubah urutan naik (ascending) atau turun (descending).
Pengecualian	- Jika data *leaderboard* kosong, sistem menampilkan pesan "Data tidak tersedia." - Jika terjadi kesalahan server atau koneksi, sistem menampilkan pesan error yang sesuai.

4.3.12 Melihat Detail Order Penjualan Produk di Dalam *Leaderboard*.

Tabel 4.14 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk melihat detail order penjualan produk di dalam *leaderboard*.

Tabel 4.14 Use Case Aplikasi Website *Sales Agent* untuk Melihat Detail Order Penjualan Produk di Dalam *Leaderboard*

Nama	Melihat Detail Order Penjualan Produk di Dalam *Leaderboard*
Kode	UC-012
Deskripsi	Use case ini menggambarkan proses *user* dalam melihat detail order penjualan produk dari seorang *sales agent* yang ditampilkan pada *leaderboard*.
Tipe	Fungsional
Pemicu	*User* mengklik kolom total penjualan sales dalam *leaderboard*.
Aktor	Admin atau *User*
Kondisi Awal	- *User* sudah login ke sistem. - Halaman data *leaderboard* tersedia.
Kondisi Akhir	Sistem menampilkan detail order penjualan produk sesuai entri yang dipilih.
Alur Kejadian Secara Normal	1. *User* membuka halaman *leaderboard*.

	2. Sistem menampilkan daftar *leaderboard*. 3. *User* memilih atau mengklik total penjualan produk dalam *leaderboard*. 4. Sistem menampilkan detail order penjualan produk terkait, termasuk informasi produk, jumlah, harga, dan jenis order.
Alur Kejadian Alternatif	- *User* dapat kembali ke halaman *leaderboard* utama dengan menekan tombol "Kembali".
Pengecualian	- Jika detail order tidak ditemukan, sistem menampilkan pesan "Data detail order tidak tersedia." - Jika terjadi kesalahan server atau koneksi, sistem menampilkan pesan error yang sesuai.

4.3.13 Melihat Performa *Sales Agent* yang Mencakup Skor Total, Ranking, Benefit, Target, Realisasi, Pencapaian, Pertumbuhan, dan Skor Tiap Produk Digital dalam Gamifikasi.

Tabel 4.15 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk melihat performa *sales agent* yang mencakup skor total, ranking, benefit, target, realisasi,

pencapaian, pertumbuhan, dan skor tiap produk digital dalam gamifikasi.

Tabel 4.15 Use Case Aplikasi Website *Sales Agent* untuk Melihat Performa *Sales Agent* yang Mencakup Skor Total, Ranking, Benefit, Target, Realisasi, Pencapaian, Pertumbuhan, dan Skor Tiap Produk Digital dalam Gamifikasi.

Nama	Melihat Performa *Sales Agent* yang Mencakup Skor Total, Ranking, Benefit, Target, Realisasi, Pencapaian, Pertumbuhan, dan Skor Tiap Produk Digital dalam Gamifikasi.
Kode	UC-013
Deskripsi	Use case ini menggambarkan proses *user* dalam melihat performa seorang *sales agent*, termasuk skor total, ranking, benefit, target, realisasi, pencapaian, pertumbuhan, dan skor tiap produk digital dalam sistem gamifikasi.
Tipe	Fungsional
Pemicu	*User* menekan tombol gamifikasi pada dashboard.
Aktor	Admin atau *User*

Kondisi Awal	- *User* sudah login ke sistem. - Data gamifikasi *sales agent* tersedia dalam sistem.
Kondisi Akhir	Sistem menampilkan performa lengkap *sales agent* sesuai data yang tersedia.
Alur Kejadian Secara Normal	1. *User* membuka halaman gamifikasi. 2. Sistem menampilkan performa *sales agent*, termasuk skor total, ranking, benefit, target, realisasi, pencapaian, pertumbuhan, dan skor tiap produk digital.
Alur Kejadian Alternatif	- *User* dapat membandingkan performa beberapa *sales agent* - *User* dapat memilih periode waktu tertentu untuk melihat performa historis.
Pengecualian	- Jika terjadi kesalahan server atau koneksi, sistem menampilkan pesan error yang sesuai.

4.3.14 Melakukan Filter Gamifikasi berdasarkan Bulan, Witel, Telda, Agency, Jenis, Kode, dan Nama *Sales Agent*.

Tabel 4.16 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk melakukan filter gamifikasi

berdasarkan bulan, witel, telda, agency, jenis, kode, dan nama *sales agent*.

Tabel 4.16 Use Case Aplikasi Website *Sales Agent* untuk Melakukan Filter Gamifikasi berdasarkan Bulan, Witel, Telda, Agency, Jenis, Kode, dan Nama *Sales Agent*.

Nama	Melihat Performa *Sales Agent* yang Mencakup Skor Total, Ranking, Benefit, Target, Realisasi, Pencapaian, Pertumbuhan, dan Skor Tiap Produk Digital dalam Gamifikasi.
Kode	UC-013
Deskripsi	Use case ini menggambarkan proses *user* dalam melihat performa seorang *sales agent*, termasuk skor total, ranking, benefit, target, realisasi, pencapaian, pertumbuhan, dan skor tiap produk digital dalam sistem gamifikasi.
Tipe	Fungsional
Pemicu	*User* menekan tombol gamifikasi pada dashboard.
Aktor	Adimin atau *User*
Kondisi Awal	- *User* sudah login ke sistem.

	- Data gamfikasi *sales agent* tersedia dalam sistem.
Kondisi Akhir	Sistem menampilkan performa lengkap *sales agen*t sesuai data yang tersedia.
Alur Kejadian Secara Normal	1. *User* membuka halaman gamfikasi. 2. Sistem menampilkan performa *sales agent*, termasuk skor total, ranking, benefit, target, realisasi, pencapaian, pertumbuhan, dan skor tiap produk digital.
Alur Kejadian Alternatif	- *User* dapat membandingkan performa beberapa *sales agent* - *User* dapat memilih periode waktu tertentu untuk melihat performa historis.
Pengecualian	- Jika terjadi kesalahan server atau koneksi, sistem menampilkan pesan error yang sesuai.

4.3.15 Mengurutkan Gamifikasi Berdasarkan Nama, Performa, Target, Realisasi, Pencapaian, Pertumbuhan, Skor, Skor Total, dan Ranking.

Tabel 4.17 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk mengurutkan gamifikasi berdasarkan

nama, performa, target, realisasi, pencapaian, pertumbuhan, skor, skor total, dan ranking.

Tabel 4.17 Use Case Aplikasi Website *Sales Agent* untuk Mengurutkan Gamifikasi Berdasarkan Nama, Performa, Target, Realisasi, Pencapaian, Pertumbuhan, Skor, Skor Total, dan Ranking.

Nama	Mengurutkan Gamifikasi Berdasarkan Nama, Performa, Target, Realisasi, Pencapaian, Pertumbuhan, Skor, Skor Total, dan Ranking.
Kode	UC-015
Deskripsi	Use case ini menggambarkan proses *user* dalam mengurutkan data gamifikasi *sales agent* berdasarkan kriteria seperti Nama, Performa, Target, Realisasi, Pencapaian, Pertumbuhan, Skor, Skor Total, dan Ranking.
Tipe	Fungsional
Pemicu	*User* menekan tombol pengurutan (*sorting*) pada kolom tabel di halaman gamifikasi.

Aktor	- Admin atau *User*
Kondisi Awal	- *User* sudah login ke sistem. - Halaman data gamifikasi tersedia dalam sistem.
Kondisi Akhir	Sistem menampilkan data gamifikasi *sales agent* sesuai urutan yang dipilih.
Alur Kejadian Secara Normal	1. *User* membuka halaman gamifikasi. 2. Sistem menampilkan data gamifikasi default (urutan standar). 3. *User* memilih kriteria pengurutan (Nama, Performa, Target, Realisasi, Pencapaian, Pertumbuhan, Skor, Skor Total, Ranking). 4. Sistem memproses permintaan pengurutan. 5. Sistem menampilkan data gamifikasi sesuai kriteria urutan yang dipilih.
Alur Kejadian Alternatif	- Jika *user* memilih lebih dari satu kriteria pengurutan, sistem mengurutkan berdasarkan kriteria terakhir. - *User* dapat memilih urutan naik (ascending) atau turun (descending).
Pengecualian	- Jika data gamifikasi kosong,

	sistem menampilkan pesan "Data tidak tersedia." - - Jika terjadi kesalahan server atau koneksi, sistem menampilkan pesan error yang sesuai.

4.3.16 Melihat Detail Order Penjualan untuk Tiap Produk di Dalam Gamifikasi.

Tabel 4.18 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk melihat detail order penjualan untuk tiap produk di dalam gamifikasi.

Tabel 4.18 Use Case Aplikasi Website *Sales Agent* untuk Melihat Detail Order Penjualan untuk Tiap Produk di Dalam Gamifikasi.

Nama	Melihat Detail Order Penjualan Tiap Produk di Dalam Gamifikasi
Kode	UC-016
Deskripsi	Use case ini menggambarkan proses *user* dalam melihat detail order penjualan untuk setiap produk digital dari seorang *sales agent* dalam modul gamifikasi, termasuk jumlah, harga, keterangan, dan jenis terkait.
Tipe	Fungsional

Pemicu	*User* menekan jumlah penjualan produk tertentu dalam halaman gamifikasi untuk melihat detail order.
Aktor	Admin atau *User*
Kondisi Awal	- *User* sudah login ke sistem. - Halaman data gamifikasi dan order penjualan tersedia.
Kondisi Akhir	Sistem menampilkan detail order penjualan produk digital sesuai yang dipilih.
Alur Kejadian Secara Normal	1. *User* membuka halaman gamifikasi. 2. Sistem menampilkan daftar *sales agent* dan skor gamifikasi. 3. *User* menekan angka penjualan produk. 4. Sistem menampilkan detail order penjualan untuk tiap jenis produk, termasuk jumlah, harga, keterangan, dan jenis terkait.
Alur Kejadian Alternatif	- *User* dapat kembali ke tampilan gamifikasi utama dengan menekan tombol “Kembali”.
Pengecualian	- Jika detail order tidak ditemukan, sistem

	menampilkan pesan "Data detail order tidak tersedia." - Jika terjadi kesalahan server atau koneksi, sistem menampilkan pesan error yang sesuai.

4.3.17 Menambahkan Target Penjualan *Sales Agent* Untuk Tiap Produk.

Tabel 4.19 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk menambahkan target penjualan *sales agent* untuk tiap produk.

Tabel 4.19 Use Case Aplikasi Website *Sales Agent* untuk Menambahkan Target Penjualan *Sales Agent* Untuk Tiap Produk.

Nama	Menambahkan Target Penjualan *Sales Agent* Untuk Tiap Produk
Kode	UC-017
Deskripsi	Use case ini menggambarkan proses admin dalam menetapkan target penjualan kepada *sales agent* untuk tiap produk digital yang dijual, sehingga dapat digunakan sebagai acuan dalam penilaian performa dan

	pencapaian.
Tipe	Fungsional
Pemicu	Admin menekan tombol sales target untuk menetapkan target penjualan untuk *sales agent* berdasarkan produk tertentu.
Aktor	Admin atau *User*
Kondisi Awal	Admin telah login ke sistem dan memiliki hak akses untuk mengatur target penjualan.
Kondisi Akhir	Target penjualan berhasil tersimpan dalam sistem dan terhubung dengan data *sales agent* serta produk terkait.
Alur Kejadian Secara Normal	1. Admin membuka halaman target penjualan *sales agent*. 2. Admin memilih *sales agen*t. 3. Admin memilih produk yang akan ditargetkan. 4. Admin memasukkan nilai target penjualan. 5. Sistem menyimpan target penjualan. 6. Sistem menampilkan notifikasi bahwa target berhasil ditambahkan.

Alur Kejadian Alternatif	Jika admin ingin mengubah target yang sudah ada, sistem menampilkan data target lama, lalu admin dapat memperbarui nilainya sebelum menyimpan.
Pengecualian	- Jika terjadi kegagalan koneksi database, sistem menampilkan pesan error.

4.3.18 Menetapkan Bobot Kriteria Tiap Produk.

Tabel 4.20 berikut merupakan tabel use case dari aplikasi website *sales agent* untuk Menambahkan target penjualan *sales agent* untuk tiap produk.

Tabel 4.20 Use Case Aplikasi Website *Sales Agent* untuk Menetapkan Bobot Kriteria Tiap Produk.

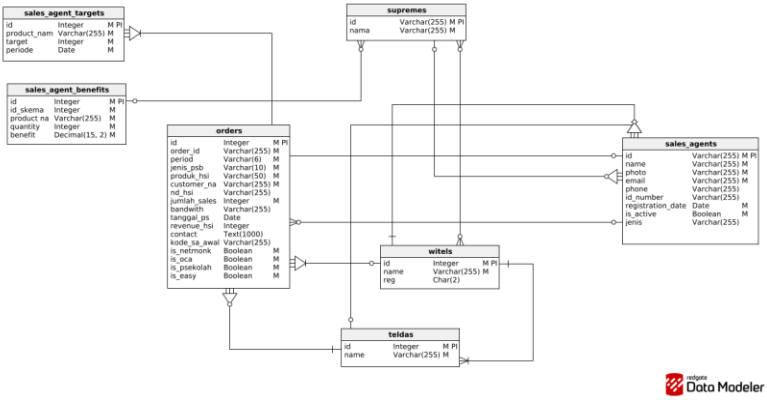
Nama	Menetapkan Bobot Kriteria Tiap Produk
Kode	UC-011
Deskripsi	Use case ini menggambarkan proses admin dalam menetapkan bobot kriteria untuk tiap produk digital yang akan digunakan dalam

	perhitungan skor performa *sales agent*.
Tipe	Fungsional
Pemicu	Admin menekan menu sales kriteria pada dashboard aplikasi web *sales agent*.
Aktor	Admin atau *User*
Kondisi Awal	Admin telah login ke sistem dan memiliki hak akses untuk mengatur bobot kriteria produk.
Kondisi Akhir	Bobot kriteria tiap produk tersimpan di sistem dan dapat digunakan dalam perhitungan performa gamifikasi *sales agent*.
Alur Kejadian Secara Normal	1. Admin membuka menu sales kriteria untuk mengatur bobot kriteria produk. 2. Admin memasukkan bobot untuk masing-masing kriteria produk. 4. Admin menyimpan pengaturan bobot. 5. Sistem menampilkan notifikasi bahwa bobot berhasil ditetapkan.
Alur Kejadian	- Jika admin ingin

Alternatif	memperbarui bobot yang sudah ada, sistem menampilkan data bobot lama, lalu admin dapat memperbaruinya sebelum menyimpan. - Jika bobot tidak seimbang (misalnya total tidak 100%), sistem memberikan peringatan dan meminta perbaikan.
Pengecualian	- Jika admin tidak mengisi nilai bobot, sistem menampilkan pesan error.- Jika input bobot melebihi batas yang ditentukan (misalnya total > 100%), sistem menandai merah untuk kriteria tersebut. - Jika terjadi error koneksi database, sistem menampilkan pesan gagal simpan.
Pengecualian	- Jika terjadi kegagalan koneksi database, sistem menampilkan pesan error.

4.4 Conceptual Data Model

Berikut adalah *conceptual* data model dari aplikasi web *sales agent*.



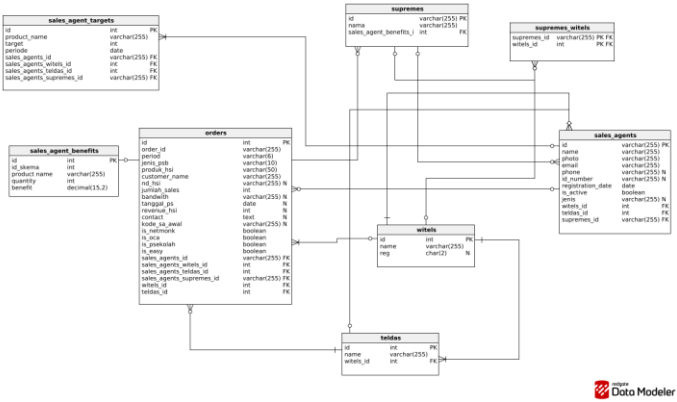
Gambar 4.5.1 Conceptual Data Model Aplikasi Web *Sales Agent*

Gambar 4.5.1 Conceptual Data Model Aplikasi Web *Sales Agent* menunjukkan tujuh tabel yang merepresentasikan data yang digunakan dalam aplikasi web *sales agent*. Tabel utama, seperti sales_agents dan orders, menyimpan informasi mengenai *sales agent* dan transaksi penjualan. Sementara itu, tabel referensi, seperti witels, teldas, supremes, sales_agent_targets, dan sales_agent_benefits, menyediakan informasi mengenai wilayah telekomunikasi, cabang atau unit wilayah, koordinator (supreme), target penjualan *sales agent*, serta benefit penjualan yang diterima. Struktur hubungan antar tabel ini memungkinkan integrasi data antara informasi *sales agent* dan transaksi penjualan produk secara menyeluruh.

4.5 *Physical* Data Model

Berikut adalah *Physical* Data Model dari aplikasi web *sales agent*.

Gambar 4.5.2 Conceptual Data Model Aplikasi Web *Sales Agent*



Gambar 4.5.2 menampilkan *Physical* Data Model (PDM) aplikasi yang mengelola data terkait *sales agent*, penjualan, witel, telda, supreme, target *sales agent*, serta benefit *sales agent* yang disimpan dalam beberapa tabel inti, seperti sales_agents, orders, witels, teldas, supremes, sales_agent_targets, dan sales_agent_benefits. Berikut merupakan penjelasan masing - masing tabel :

1. Tabel sales_agents : Menyediakan informasi *sales agent* yang mencakup id, nama, foto, email, nomor telefon, nomor KTP, tanggal registrasi, status keaktifan, jenis sales, witel, telda, dan supreme.
2. Tabel orders : Menyediakan informasi penjualan *sales agent* yang mecakup id, period, jenis, produk, nama customer, dan lain -lain.

3. Tabel witels : Menyimpan informasi mengenai wilayah telekomunikasi.
4. Tabel teldas : Menyimpan informasi mengenai daerah telekomunikasi.
5. Tabel sales_agents_targets : Menyediakan informasi target penjualan sales untuk setiap produk.
6. Tabel sales_agents_benefits : Menyediakan informasi benefit penjualan sales untuk setiap produk.

Hubungan antar tabel dijaga melalui *foreign key* yang memastikan integritas data, misalnya antara tabel sales_agents dengan witels, teldas, dan supremes, serta tabel orders dengan sales_agents, witels, dan teldas.

[Halaman ini sengaja dikosongkan]

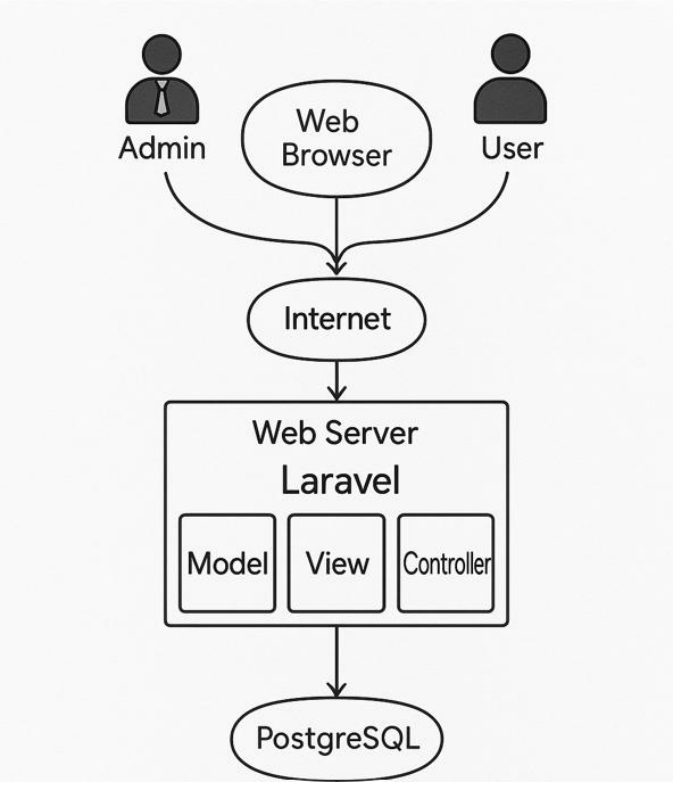
BAB V
IMPLEMENTASI SISTEM

Bab ini membahas mengenai implementasi sistem yang telah kami kembangkan. Proses implementasi dibagi menjadi beberapa bagian, yaitu implementasi model, view, controller, serta antarmuka aplikasi website *Sales Agent* berbasis Laravel dengan menggunakan database PostgreSQL.

5.1 Implementasi Sistem

5.2 Implementasi Arsitektur Sistem

Pada bagian ini akan digambarkan arsitektur sistem dari aplikasi web *Sales Agent* berbasis gamifikasi dan *leaderboard* yang dikembangkan menggunakan *framework* Laravel dengan pola arsitektur *Model-View-Controller* (MVC). Adapun diagram arsitektur sistem untuk aplikasi web *Sales Agent* dapat dilihat pada Gambar 5.2.1.



Gambar 5.2.1 Arsitektur Sistem Aplikasi Web *Sales Agent*

Pada Gambar 5.2.1 ditampilkan diagram arsitektur sistem untuk aplikasi web *Sales Agent*. Diagram ini memperlihatkan bahwa terdapat dua jenis pengguna utama, yaitu admin dan user, yang mengakses aplikasi melalui *web browser* menggunakan koneksi internet.

Permintaan dari pengguna diteruskan ke web server yang menjalankan *framework* Laravel. Arsitektur Laravel berbasis MVC memungkinkan pemisahan tanggung jawab antara:

1. Model, yang berfungsi untuk berinteraksi dengan database PostgreSQL, mengelola data seperti informasi *sales agent*, penjualan, target, benefit, dan hasil pencapaian.
2. View, yang menampilkan antarmuka web kepada pengguna berupa halaman *dashboard*, *leaderboard*, dan fitur gamifikasi.
3. Controller, yang bertugas mengatur alur logika aplikasi, menghubungkan antara *View* dan *Model*, serta menangani setiap permintaan dari pengguna.

Data yang dimasukkan atau dimodifikasi melalui antarmuka web akan dikirim ke *Controller*, diproses, lalu disimpan ke dalam database PostgreSQL yang berperan sebagai pusat penyimpanan data.

5.2.1 Implementasi Model

Berikut adalah implementasi *Model* untuk aplikasi web *Sales Agent* berbasis gamifikasi dan *leaderboard*. Pada bagian ini dijelaskan bagaimana struktur dan fungsionalitas *Model* dirancang untuk mengelola data yang digunakan oleh sistem. Adapun *Model* yang digunakan dalam aplikasi web *Sales Agent* ini adalah sebagai berikut:

5.2.1.1 Telda

Potongan kode implementasi model Supreme dapat dilihat pada Gambar 5.2.1.1.

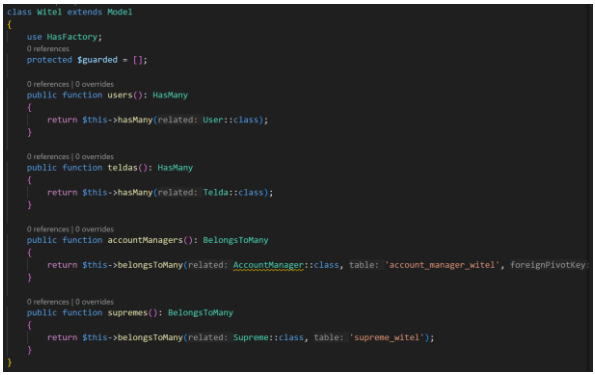
```php
class Telda extends Model
{
    use HasFactory;

    protected $guarded = [];

    public function witel(): BelongsTo
    {
        return $this->belongsTo(Witel::class);
    }

    public function stos(): HasMany
    {
        return $this->hasMany(Sto::class);
    }

    public function visitPlans(): HasMany
    {
        return $this->hasMany(VisitPlan::class);
    }
}
```

Gambar 5.2.1.1 Contoh Implementasi Model Telda

Model Telda digunakan untuk merepresentasikan data wilayah Telda dalam sistem. Model ini menunjukkan bahwa setiap Telda terhubung dengan satu Witel, serta dapat memiliki banyak STO dan banyak rencana kunjungan (*Visit Plan*). Dengan relasi tersebut, model Telda berperan penting dalam menghubungkan data wilayah dengan data operasional dan perencanaan kunjungan pada aplikasi web *Sales Agent*.

5.2.1.2 Witel

Potongan kode implementasi model Witel dapat dilihat pada Gambar 5.2.1.2.



Gambar 5.2.1.2 Contoh Implementasi Model Telda

Model Witel (Wilayah Telekomunikasi) digunakan untuk merepresentasikan struktur organisasi regional tingkat tinggi dalam sistem. Model ini menunjukkan bahwa satu Witel dapat memiliki banyak *User* (pengguna/karyawan), terhubung dengan banyak wilayah Telda, dan juga terkait dengan banyak Account Manager dan Supreme (peran pengawas atau manajemen). Dengan relasi-relasi tersebut, model Witel berfungsi sebagai pusat pengelompokan (hub) utama yang vital, menghubungkan data wilayah geografis (Telda) dengan data sumber daya manusia dan struktur manajemen yang bertanggung jawab atas wilayah tersebut.

5.2.1.3 Supreme

Potongan kode implementasi model Supreme dapat dilihat pada Gambar 5.2.1.3.

```
class Supreme extends Model
{

    protected $keyType = 'string';
    public $incrementing = false;

    protected $fillable = [
        'id',
        'name',
        'skema_benefit',
    ];


    public function witels(): BelongsToMany
    {
        return $this->belongsToMany(Witel::class, 'supreme_witel');
    }


    public function salesAgentBenefits(): HasMany
    {
        return $this->hasMany(SalesAgentBenefit::class, 'id_skema', 'skema_benefit');
    }
```

Gambar 5.2.1.3 Contoh Implementasi Model Supreme

Model Supreme digunakan untuk merepresentasikan data skema benefit dalam sistem. Model ini memiliki relasi *many-to-many* dengan Witel melalui tabel perantara *supreme_witel*, serta relasi *one-to-many* dengan data benefit *sales agent* berdasarkan skema benefit yang dimiliki. Dengan relasi tersebut, model Supreme berfungsi sebagai penghubung antara wilayah (Witel) dan skema benefit yang diterapkan pada setiap *sales agent* dalam aplikasi web *Sales Agent*.

5.2.1.4 Sales Agent

Potongan kode implementasi model Sales Agent dapat dilihat pada Gambar 5.2.1.4.

```php
class SalesAgent extends Model
{
    0 references
    protected $keyType = 'string';
    0 references
    public $incrementing = false;

    0 references
    protected $fillable = [
        'id',
        'name',
        'witel_id',
        'telda_id',
        'supreme_id',
        'email',
        'phone',
        'id_number',
        'registration_date',
        'is_active',
        'jenis',
        'photo',
    ];

    0 references
    protected $casts = [
        'registration_date' => 'date',
        'is_active' => 'boolean',
    ];

    0 references | 0 overrides
    public function witel(): BelongsTo
    {
        return $this->belongsTo(related: Witel::class);
    }

    0 references | 0 overrides
    public function supreme(): BelongsTo
    {
```

Gambar 5.2.1.4 Contoh Implementasi Model Sales Agent

Model SalesAgent adalah inti dari sistem, berperan untuk merepresentasikan setiap individu agen penjualan. Model ini menyimpan data penting agen seperti informasi kontak, status keaktifan (*is_active*), dan detail identitas.

5.2.1.5 Order

Potongan kode implementasi model order dapat dilihat pada Gambar 5.2.1.5.

```
8   class Order extends Model
10
11      protected $fillable = [
12          'order_id',
13          'period',
14          'jenis_psb',
15          'produk_hsi',
16          'witel_id',
17          'telda_id',
18          'customer_name',
19          'nd_hsi',
20          'jumlah_sales',
21          'bandwith',
22          'tanggal_ps',
23          'revenue_hsi',
24          'contact',
25          'kode_sa',
26          'kode_sa_awal',
27          'is_netmonk',
28          'is_oca',
29          'is_psekolah',
30          'is_easy',
31      ];
```

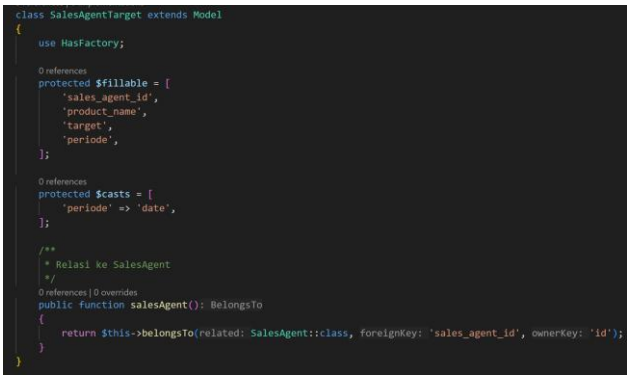
Gambar 5.2.1.5 Contoh Implementasi Model Order

Kode ini mendefinisikan model order yang merepresentasikan data pesanan. Model ini

menentukan kolom yang dapat diisi, mengonversi tanggal otomatis ke tipe *date*, serta membangun relasi belongsTo dengan model Witel, Telda, dan relasi SalesAgent untuk menghubungkan setiap pesanan dengan data wilayah dan agen penjualan terkait.

5.2.1.6 Sales Agent Target

Potongan kode implementasi model Sales Agent Target dapat dilihat pada Gambar 5.2.1.6.



Gambar 5.2.1.6 Contoh Implementasi Model Sales Agent Target

Secara fungsional, SalesAgentTarget berperan vital dalam modul pelaporan dan manajemen kinerja. Dengan menghubungkan target yang ditetapkan ke Agen Penjualan, sistem dapat melakukan perhitungan otomatis untuk membandingkan hasil aktual (dari data *Orders* di Model SalesAgent) dengan target yang direncanakan,

sehingga memudahkan manajer (Supreme) dalam evaluasi dan insentif.

5.2.1.7 Sales Agent Benefit

Potongan kode implementasi model Sales Agent Benefit dapat dilihat pada Gambar 5.2.1.7.

```php
class SalesAgentBenefit extends Model
{
    protected $fillable = [
        'id_skema',
        'product_name',
        'quantity',
        'benefit',
    ];

    /**
     * Relasi ke Supreme (pakai skema_benefit).
     */
    public function supreme(): BelongsTo
    {
        return $this->belongsTo(Supreme::class, 'id_skema', 'skema_benefit');
    }
}
```

Gambar 5.2.1.7 Contoh Implementasi Model Sales Agent Benefit

Model SalesAgentBenefit digunakan untuk merepresentasikan data *benefit* atau penghargaan yang diterima oleh *sales agent* berdasarkan skema yang berlaku. Model ini memiliki relasi *belongsTo* dengan model Supreme, sehingga setiap *benefit* terkait langsung dengan skema benefit tertentu. Dengan demikian, model ini berperan dalam mengelola dan menghubungkan data benefit produk yang diperoleh *sales agent* dengan skema reward yang telah ditetapkan dalam aplikasi web *Sales Agent*.

5.2.1.8 Sales Criteria

Potongan kode implementasi model Sales Criteria dapat dilihat pada Gambar 5.2.1.8.

```
class SalesCriteria extends Model
{
    0 references
    protected $table = 'sales_criteria';
    0 references
    protected $fillable = ['nama', 'bobot', 'score_max'];
    0 references
    public $timestamps = false;
}
```

Gambar 5.2.1.8 Contoh Implementasi Model Sales Criteria

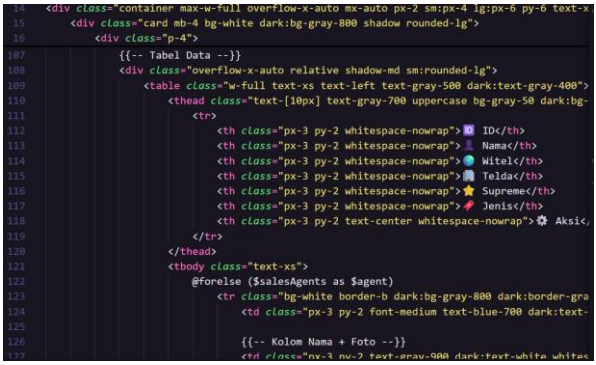
Model SalesCriteria digunakan untuk menyimpan dan mendefinisikan kriteria yang digunakan dalam penilaian atau evaluasi kinerja penjualan.

5.2.2 Implementasi *Views*

Pada bagian ini dibahas mengenai implementasi views dari aplikasi *Sales Agent*, yang berfungsi sebagai sisi *frontend* dari sistem. Bagian ini menampilkan rancangan antarmuka pengguna yang dibangun menggunakan template Konrix dengan bantuan Tailwind CSS, sehingga menghasilkan tampilan yang responsif, modern, dan mudah digunakan oleh pengguna. Berikut merupakan implementasi dari views untuk halaman pada aplikasi *Sales Agent*.

5.2.2.1 Halaman *Index Sales Agent*

Potongan kode implementasi views halaman *create new sales Agent* dapat dilihat pada Gambar 5.2.2.1.



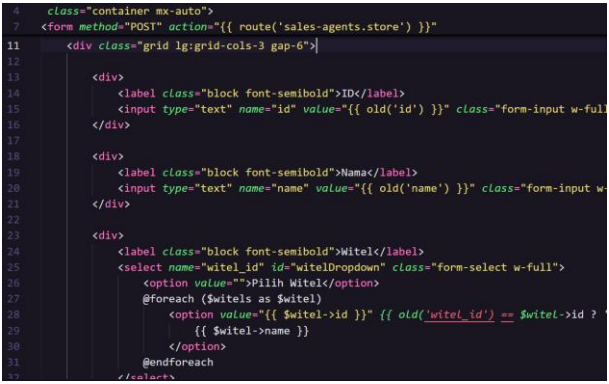
```
<div class="container max-w-full overflow-x-auto mx-auto px-2 sm:px-4 lg:px-6 py-6 text-x
    <div class="card mb-4 bg-white dark:bg-gray-800 shadow rounded-lg">
        <div class="p-4">
            {{-- Tabel Data --}}
            <div class="overflow-x-auto relative shadow-md sm:rounded-lg">
                <table class="w-full text-xs text-left text-gray-500 dark:text-gray-400">
                    <thead class="text-[10px] text-gray-700 uppercase bg-gray-50 dark:bg-
                        <tr>
                            <th class="px-3 py-2 whitespace-nowrap">🆔 ID</th>
                            <th class="px-3 py-2 whitespace-nowrap">👤 Nama</th>
                            <th class="px-3 py-2 whitespace-nowrap">🌐 Witel</th>
                            <th class="px-3 py-2 whitespace-nowrap">🏙️ Telda</th>
                            <th class="px-3 py-2 whitespace-nowrap">⭐ Supreme</th>
                            <th class="px-3 py-2 whitespace-nowrap">📌 Jenis</th>
                            <th class="px-3 py-2 text-center whitespace-nowrap">⚙️ Aksi</
                        </tr>
                    </thead>
                    <tbody class="text-xs">
                        @forelse ($salesAgents as $agent)
                            <tr class="bg-white border-b dark:bg-gray-800 dark:border-gra
                                <td class="px-3 py-2 font-medium text-blue-700 dark:text-

                                {{-- Kolom Nama + Foto --}}
                                <td class="px-3 py-2 text-gray-900 dark:text-white whites
```

Gambar 5.2.2.1 Potongan Kode Halaman *Index Sales Agent*

Kode ini merupakan tampilan halaman daftar *Sales Agent* yang menampilkan data dalam tabel lengkap dengan fitur filter, pencarian, dan aksi seperti lihat, edit, serta hapus data. Tampilan menggunakan Tailwind CSS dan beberapa plugin seperti Flatpickr dan Tom Select untuk mempercantik antarmuka dan mempermudah interaksi pengguna.

5.2.2.2 Halaman *Create New Sales Agent*

Potongan kode implementasi views halaman *create new sales agent* dapat dilihat pada Gambar 5.2.2.2.

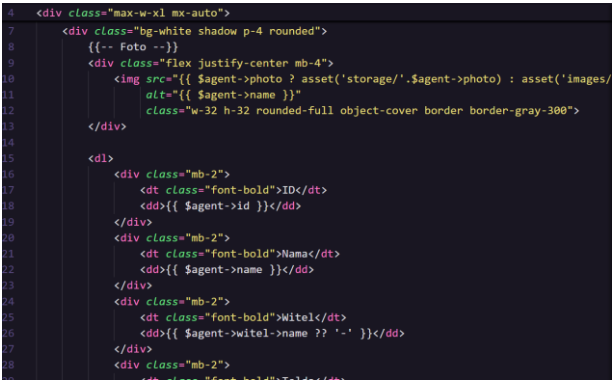


Gambar 5.2.2.2 Potongan Kode Halaman *Create New Sales Agent*

Kode tersebut merupakan halaman form Laravel untuk menambah data *sales agent,* yang menampilkan berbagai input dan mengirim data ke server melalui metode POST dengan tampilan rapi menggunakan Tailwind CSS.

5.2.2.3 Halaman *Show Sales Agent*

Potongan kode implementasi views halaman *show sales agent* dapat dilihat pada Gambar 5.2.2.3.



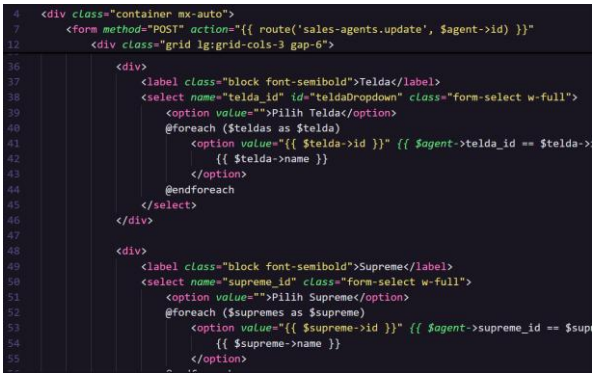
```
<div class="max-w-xl mx-auto">
    <div class="bg-white shadow p-4 rounded">
        {{-- Foto --}}
        <div class="flex justify-center mb-4">
            <img src="{{ $agent->photo ? asset('storage/'.$agent->photo) : asset('images/
                alt="{{ $agent->name }}"
                class="w-32 h-32 rounded-full object-cover border border-gray-300">
        </div>

        <dl>
            <div class="mb-2">
                <dt class="font-bold">ID</dt>
                <dd>{{ $agent->id }}</dd>
            </div>
            <div class="mb-2">
                <dt class="font-bold">Nama</dt>
                <dd>{{ $agent->name }}</dd>
            </div>
            <div class="mb-2">
                <dt class="font-bold">Witel</dt>
                <dd>{{ $agent->witel->name ?? '-' }}</dd>
            </div>
            <div class="mb-2">
```

Gambar 5.2.2.3 Potongan Kode Halaman *Show Sales Agent*

Kode ini menampilkan halaman detail *sales agent*, berisi informasi lengkap seperti foto, identitas, kontak, dan status dalam tampilan sederhana menggunakan Tailwind CSS, serta menyediakan tombol untuk kembali ke daftar data.

5.2.2.4 Halaman Edit *Sales Agent*

Potongan kode implementasi views halaman edit *Sales Agent* dapat dilihat pada Gambar 5.2.2.4.



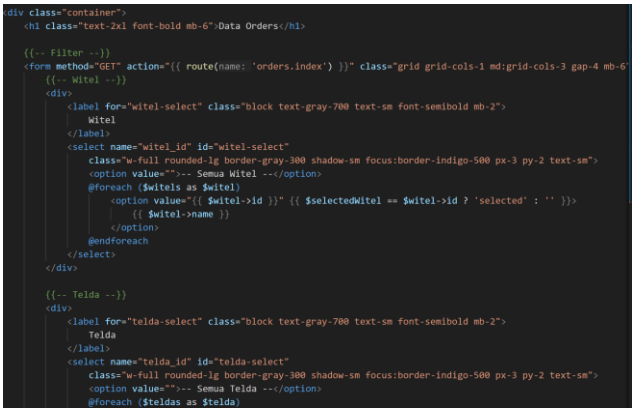
```
<div class="container mx-auto">
    <form method="POST" action="{{ route('sales-agents.update', $agent->id) }}"
        <div class="grid lg:grid-cols-3 gap-6">

            <div>
                <label class="block font-semibold">Telda</label>
                <select name="telda_id" id="teldaDropdown" class="form-select w-full">
                    <option value="">Pilih Telda</option>
                    @foreach ($teldas as $telda)
                        <option value="{{ $telda->id }}" {{ $agent->telda_id == $telda->
                            {{ $telda->name }}
                        </option>
                    @endforeach
                </select>
            </div>

            <div>
                <label class="block font-semibold">Supreme</label>
                <select name="supreme_id" class="form-select w-full">
                    <option value="">Pilih Supreme</option>
                    @foreach ($supremes as $supreme)
                        <option value="{{ $supreme->id }}" {{ $agent->supreme_id == $sup
                            {{ $supreme->name }}
                        </option>
```

Gambar 5.2.2.4 Potongan Kode Halaman Edit *Sales Agent*

Kode ini merupakan halaman form untuk mengedit data *Sales Agent*. Halaman ini menampilkan data yang sudah ada agar dapat diperbarui melalui form yang rapi menggunakan Tailwind CSS, dengan metode PUT ke route pembaruan data. Selain input teks dan pilihan, form ini juga mendukung unggah foto serta pemuatan dinamis data Telda berdasarkan Witel melalui JavaScript.

5.2.2.5 Halaman *Index Orders*

Potongan kode implementasi views halaman *orders index* dapat dilihat pada Gambar 5.2.2.5

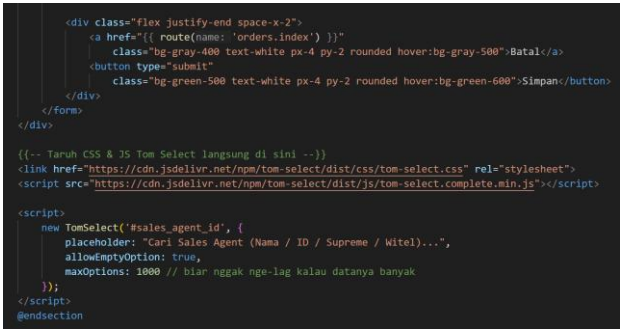


Gambar 5.2.2.5 Potongan Kode Halaman *Index Orders*

Kode ini menampilkan halaman Daftar Data Orders menggunakan Tailwind CSS, dilengkapi dengan fungsionalitas filter berdasarkan Witel dan Telda, serta fitur pencarian untuk Sales Agent dan Order ID. Data *orders* disajikan dalam bentuk tabel yang lengkap dan dapat di-*scroll*, mencakup detail order, penjualan, pelanggan, dan lokasi, serta menyediakan tombol Edit untuk setiap order.

5.2.2.6 Halaman Edit *Orders*

Potongan kode implementasi views halaman edit *orders* dapat dilihat pada Gambar 5.2.2.6.



Gambar 5.2.2.6 Potongan Kode Halaman Edit *Orders*

Kode ini menampilkan halaman ringkas untuk mengedit *sales agent* pada sebuah *Order* spesifik. Formulir menggunakan Tom Select (dengan fitur pencarian) untuk memilih *sales agent* baru berdasarkan Nama, ID, Supreme, atau Witel, lalu menyediakan tombol simpan untuk memperbarui data (*PUT*) dan tombol batal untuk kembali ke daftar *orders*.

5.2.2.7 Halaman Index Leaderboard

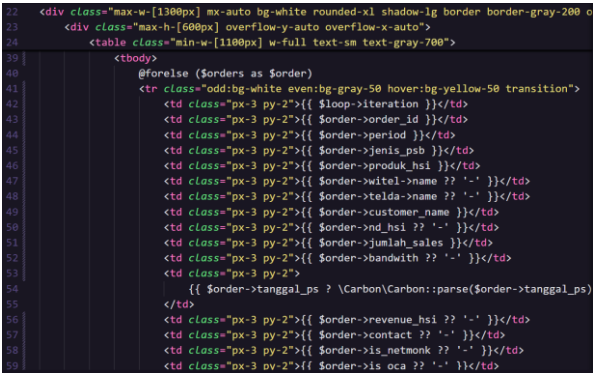
Potongan kode implementasi views halaman *index leaderboard* dapat dilihat pada Gambar 5.2.2.7.

Gambar 5.2.2.7 Potongan Kode Halaman *Index Leaderboard*

Potongan kode ini menampilkan halaman leaderboard penjualan sales. Halaman ini berisi form filter (tanggal, produk, witel, telda, agency, jenis sales, dan pencarian) serta tabel peringkat performa sales. Data bisa diurutkan berdasarkan kolom tertentu dan menampilkan hasil dinamis sesuai filter. Tampilan dibuat responsif dan interaktif dengan Tailwind CSS dan Tom Select.

5.2.2.8 Halaman Detail *Leaderboard*

Potongan kode implementasi views halaman detail *leaderboard* dapat dilihat pada Gambar 5.2.2.8.



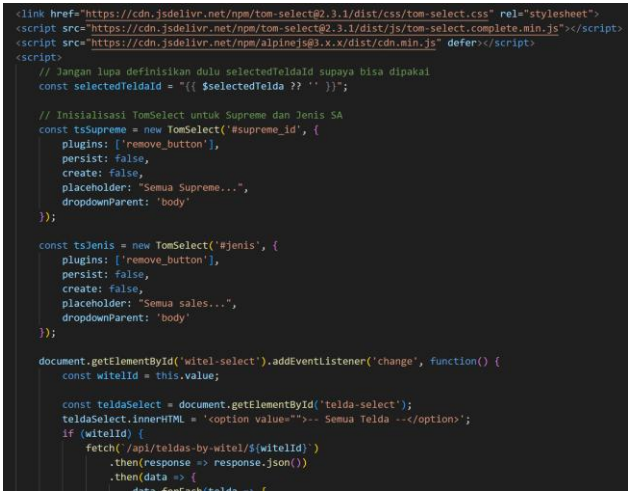
```
<div class="max-w-[1300px] mx-auto bg-white rounded-xl shadow-lg border border-gray-200 o
    <div class="max-h-[600px] overflow-y-auto overflow-x-auto">
        <table class="min-w-[1100px] w-full text-sm text-gray-700">
            <tbody>
                @forelse ($orders as $order)
                <tr class="odd:bg-white even:bg-gray-50 hover:bg-yellow-50 transition">
                    <td class="px-3 py-2">{{ $loop->iteration }}</td>
                    <td class="px-3 py-2">{{ $order->order_id }}</td>
                    <td class="px-3 py-2">{{ $order->period }}</td>
                    <td class="px-3 py-2">{{ $order->jenis_psb }}</td>
                    <td class="px-3 py-2">{{ $order->produk_hsi }}</td>
                    <td class="px-3 py-2">{{ $order->witel->name ?? '-' }}</td>
                    <td class="px-3 py-2">{{ $order->telda->name ?? '-' }}</td>
                    <td class="px-3 py-2">{{ $order->customer_name }}</td>
                    <td class="px-3 py-2">{{ $order->nd_hsi ?? '-' }}</td>
                    <td class="px-3 py-2">{{ $order->jumlah_sales }}</td>
                    <td class="px-3 py-2">{{ $order->bandwith ?? '-' }}</td>
                    <td class="px-3 py-2">
                        {{ $order->tanggal_ps ? \Carbon\Carbon::parse($order->tanggal_ps)
                    </td>
                    <td class="px-3 py-2">{{ $order->revenue_hsi ?? '-' }}</td>
                    <td class="px-3 py-2">{{ $order->contact ?? '-' }}</td>
                    <td class="px-3 py-2">{{ $order->is_netmonk ?? '-' }}</td>
                    <td class="px-3 py-2">{{ $order->is_oca ?? '-' }}</td>
```

Gambar 5.2.2.8 Potongan Kode Halaman Detail *Leaderboard*

Potongan kode ini menampilkan halaman detail order untuk *Sales Agent* di *leaderboard*, berisi daftar pesanan lengkap dengan informasi seperti periode, produk, pelanggan, revenue, dan sales terkait. Data ditampilkan dalam tabel rapi menggunakan Tailwind CSS, dengan tombol kuning untuk kembali ke halaman sebelumnya.

5.2.2.9 Halaman *Index Gamification*

Potongan kode implementasi views halaman *index gamification* dapat dilihat pada Gambar 5.2.2.9.

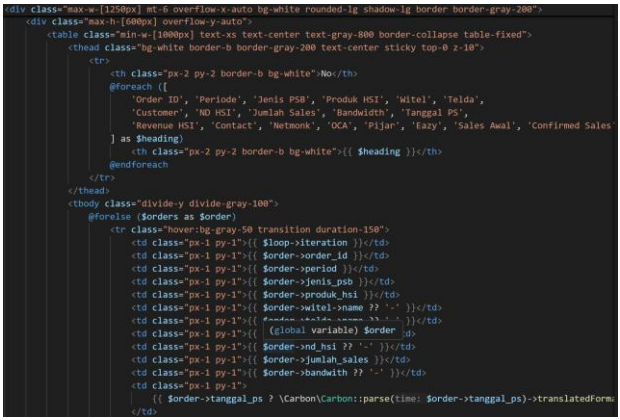


Gambar 5.2.2.9 Potongan Kode Halaman *Index Gamification*

Halaman ini menampilkan Gamification Sales Leaderboard yang detail, memungkinkan pengguna memfilter dan melihat peringkat performa *Sales Agent* berdasarkan Bulan, Witel, Telda, *Agency* (Supreme), dan Jenis Channel. Data performa disajikan dalam tabel besar dengan *header* dan *kolom* yang *sticky*, merinci skor (*Target, Real, Ach, Growth, Score*) untuk enam kriteria berbeda (*HSI, WMS, Netmonk, OCA, Pijar, Eazy*) lengkap dengan bobotnya, serta menampilkan Total Score, Ranking, dan Total Benefit, dengan dukungan fungsionalitas sorting dan pemilihan *multi-select* menggunakan Tom Select.

5.2.2.10 Halaman Detail *Gamification*

Potongan kode implementasi views halaman *detail gamification* dapat dilihat pada Gambar 5.2.2.10.

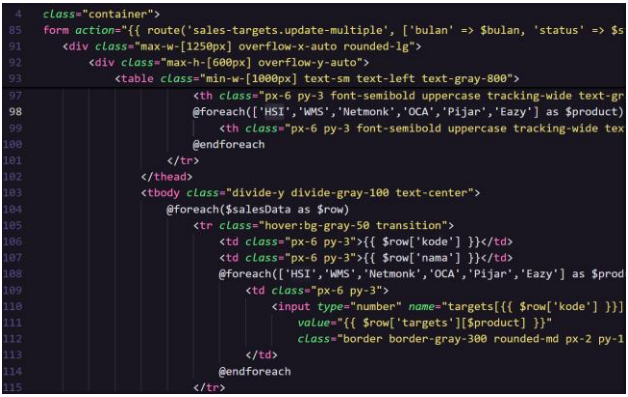


Gambar 5.2.2.10 Potongan Kode Halaman Detail *Gamification*

Halaman ini menampilkan detail daftar *order* untuk satu *Sales Agent* tertentu, difilter berdasarkan kriteria performa spesifik dalam periode tanggal tertentu. Data disajikan dalam tabel lengkap yang dapat di-*scroll*, merinci informasi order, pelanggan, produk, lokasi, status verifikasi kriteria (*Netmonk, OCA, Pijar, Eazy*), serta Sales Awal dan *Confirmed Sales*, dilengkapi tombol untuk kembali ke halaman sebelumnya.

5.2.2.11 Halaman *Index Sales Agent* Target

Potongan kode implementasi views halaman index *sales agent* target dapat dilihat pada Gambar 5.2.2.11.



```
4   class="container">
85  form action="{{ route('sales-targets.update-multiple', ['bulan' => $bulan, 'status' => $s
91      <div class="max-w-[1250px] overflow-x-auto rounded-lg">
92          <div class="max-h-[600px] overflow-y-auto">
93              <table class="min-w-[1000px] text-sm text-left text-gray-800">
97                      <th class="px-6 py-3 font-semibold uppercase tracking-wide text-gr
98                      @foreach(['HSI','WMS','Netmonk','OCA','Pijar','Eazy'] as $product)
99                          <th class="px-6 py-3 font-semibold uppercase tracking-wide tex
100                     @endforeach
101                 </tr>
102             </thead>
103             <tbody class="divide-y divide-gray-100 text-center">
104                 @foreach($salesData as $row)
105                     <tr class="hover:bg-gray-50 transition">
106                         <td class="px-6 py-3">{{ $row['kode'] }}</td>
107                         <td class="px-6 py-3">{{ $row['nama'] }}</td>
108                         @foreach(['HSI','WMS','Netmonk','OCA','Pijar','Eazy'] as $prod
109                             <td class="px-6 py-3">
110                                 <input type="number" name="targets[{{ $row['kode'] }}]
111                                     value="{{ $row['targets'][$product] }}"
112                                     class="border border-gray-300 rounded-md px-2 py-1
113                             </td>
114                         @endforeach
115                     </tr>
```

Gambar 5.2.2.11 Potongan Kode Halaman *Index Sales Agent* Target

Potongan kode ini menampilkan halaman daftar target sales dengan layout vertical. Halaman ini menyediakan fitur untuk menyalin target dari bulan lain, memfilter data berdasarkan bulan, status, dan pencarian, serta menampilkan tabel target penjualan tiap produk yang bisa langsung diedit.

5.2.2.12 Halaman *Index Sales Agent Benefit*

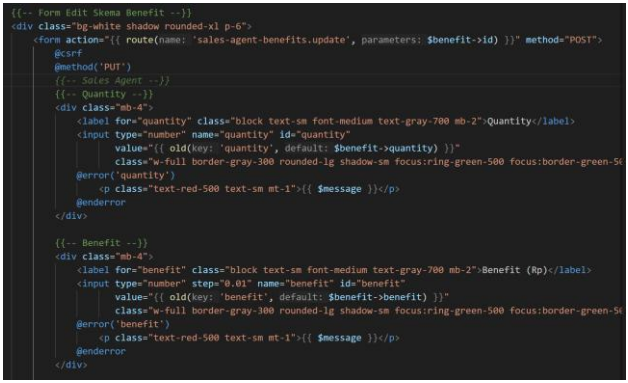
Potongan kode implementasi views halaman *index sales agent benefit* dapat dilihat pada Gambar 5.2.2.12.

Gambar 5.2.2.12 Potongan Kode Halaman *Index Sales Agent Benefit*

Kode ini menampilkan halaman daftar skema benefit untuk *sales agent* dalam format yang ringkas dan interaktif, didukung oleh Tailwind CSS. Daftar benefit dikelompokkan berdasarkan Nomor Skema (Skema ID) yang berfungsi sebagai tombol toggle (accordion) untuk menampilkan atau menyembunyikan detail di dalamnya. Setiap skema kemudian dikelompokkan lagi berdasarkan nama produk. Setiap rule benefit yang detail (menampilkan kuantitas dan besaran benefit dalam Rupiah) menyediakan tombol edit dan delete untuk manajemen data. Selain itu, terdapat tombol "Add Skema" di bagian atas untuk mengarahkan pengguna membuat rule benefit yang baru.

5.2.2.13 Halaman Edit *Sales Agent Benefit*

Potongan kode implementasi views halaman edit *sales agent benefit* dapat dilihat pada Gambar 5.2.2.13.



```
{{-- Form Edit Skema Benefit --}}
<div class="bg-white shadow rounded-xl p-6">
    <form action="{{ route(name: 'sales-agent-benefits.update', parameters: $benefit->id) }}" method="POST">
        @csrf
        @method('PUT')
        {{-- Sales Agent --}}
        {{-- Quantity --}}
        <div class="mb-4">
            <label for="quantity" class="block text-sm font-medium text-gray-700 mb-2">Quantity</label>
            <input type="number" name="quantity" id="quantity"
                   value="{{ old(key: 'quantity', default: $benefit->quantity) }}"
                   class="w-full border-gray-300 rounded-lg shadow-sm focus:ring-green-500 focus:border-green-5
            @error('quantity')
                <p class="text-red-500 text-sm mt-1">{{ $message }}</p>
            @enderror
        </div>

        {{-- Benefit --}}
        <div class="mb-4">
            <label for="benefit" class="block text-sm font-medium text-gray-700 mb-2">Benefit (Rp)</label>
            <input type="number" step="0.01" name="benefit" id="benefit"
                   value="{{ old(key: 'benefit', default: $benefit->benefit) }}"
                   class="w-full border-gray-300 rounded-lg shadow-sm focus:ring-green-500 focus:border-green-5
            @error('benefit')
                <p class="text-red-500 text-sm mt-1">{{ $message }}</p>
            @enderror
        </div>
```

Gambar 5.2.2.13 Potongan Kode Halaman *Edit Sales Agent Benefit*

Kode ini menampilkan halaman sederhana untuk mengedit detail skema *benefit* yang sudah ada. Halaman ini menggunakan Tailwind CSS untuk desain yang bersih dan berfokus pada formulir. Form tersebut memungkinkan pengguna memperbarui dua nilai utama dari sebuah rule *benefit*: *quantity* (jumlah) dan besaran *benefit* dalam Rupiah. Data yang ada akan dimuat ke dalam *field*, dan *form* akan mengirim pembaruan menggunakan metode PUT ke rute sales-agent-benefits.update. Terdapat validasi error message di bawah setiap field, serta tombol Update untuk menyimpan perubahan dan tombol Back untuk kembali ke daftar skema.

5.2.2.14 Halaman Create *Sales Agent Benefit*

Potongan kode implementasi views halaman create *sales agent benefit* dapat dilihat pada Gambar 5.2.2.14.

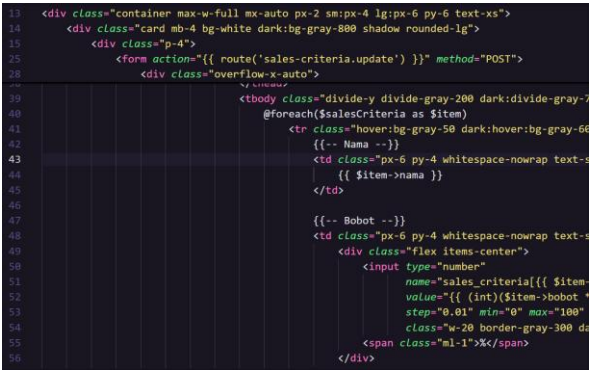
```
<form method="POST" action="{{ route(name: 'sales-agent-benefits.store') }}"
      class="bg-white p-6 rounded shadow max-w-6xl mx-auto">
    @csrf

    <div class="mb-6">
        <label for="id_skema" class="block mb-2 font-medium">No Skema</label>
        <input type="number" name="id_skema" id="id_skema"
               class="w-full border rounded px-3 py-2 max-w-xs" required>
    </div>

    <div class="overflow-x-auto">
        <table class="min-w-full border border-gray-300 text-sm">
            <thead class="bg-gray-100">
                <tr>
                    <th class="border px-3 py-2">No</th>
                    <th class="border px-3 py-2">HSI</th>
                    <th class="border px-3 py-2">WMS</th>
                    <th class="border px-3 py-2">Netmonk</th>
                    <th class="border px-3 py-2">OCA</th>
                    <th class="border px-3 py-2">Pijar</th>
                    <th class="border px-3 py-2">Eazy</th>
                </tr>
            </thead>
            <tbody>
```

Gambar 5.2.2.14 Potongan Kode Halaman *Create Sales Agent Benefit*

Kode ini menampilkan formulir untuk Menambah *skema benefit* baru yang besar dan detail. Pengguna diwajibkan menginput nomor skema (id_skema) dan mengisi nilai *benefit* (dalam Rp) dalam format tabel matrik 30x6, di mana baris mewakili kuantitas (1 hingga 30) dan kolom mewakili jenis produk (HSI, WMS, Netmonk, OCA, Pijar, Eazy). Form ini dilengkapi tombol *save* untuk menyimpan skema baru dan *back* untuk kembali.

5.2.2.15 Halaman *Index Sales Agent* Kriteria

Potongan kode implementasi views halaman index *sales agent* kriteria dapat dilihat pada Gambar 5.2.2.15.

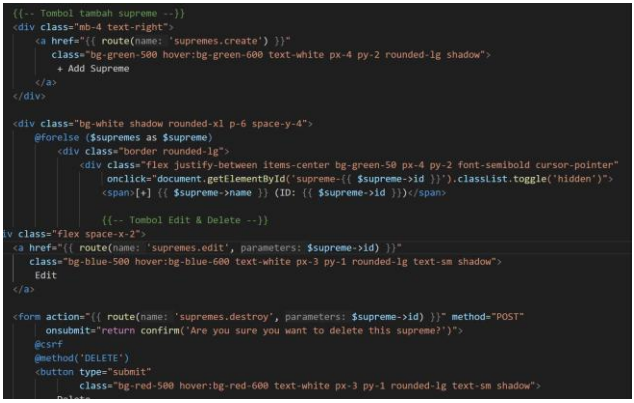


Gambar 5.2.2.15 Potongan Kode Halaman *Index Sales Agent* Kriteria

Potongan kode ini menampilkan halaman pengaturan kriteria penilaian sales. Halaman ini memuat tabel berisi daftar kriteria, bobot, dan skor maksimum yang bisa diedit langsung, lalu dihitung total bobotnya secara otomatis dengan JavaScript.

5.2.2.16 Halaman *Index Supreme*

Potongan kode implementasi views halaman *index supreme* dapat dilihat pada Gambar 5.2.2.16.



Gambar 5.2.2.16 Potongan Kode Halaman *Index Supreme*

Kode ini menampilkan halaman daftar *supreme* (*agency*/manajemen) dalam bentuk accordion interaktif menggunakan Tailwind CSS. Setiap Supreme disajikan dalam blok yang dapat diklik untuk menampilkan atau menyembunyikan detailnya. Header setiap blok menampilkan nama dan ID *supreme*, dan dilengkapi dengan tombol edit dan *delete* untuk manajemen data. Bagian detail yang tersembunyi berisi informasi skema *benefit* yang digunakan *supreme* tersebut dan menampilkan rincian benefit berupa tabel produk, kuantitas, dan nilai benefit (Rp). Terdapat juga tombol *add supreme* di bagian atas untuk menambahkan data *supreme* baru.

5.2.2.17 Halaman *Edit Supreme*

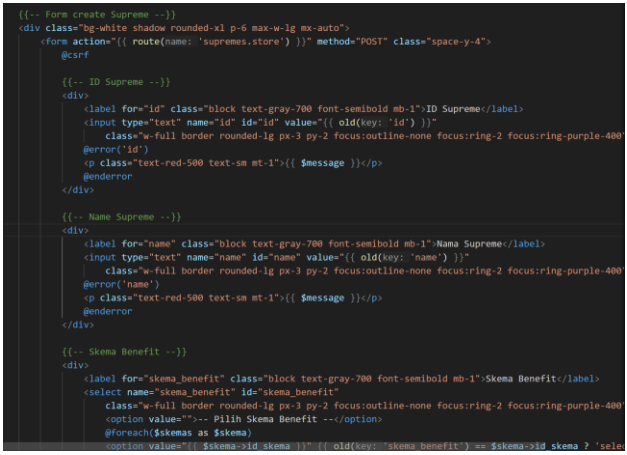
Potongan kode implementasi views halaman edit *supreme* dapat dilihat pada Gambar 5.2.2.17.

Gambar 5.2.2.17 Potongan Kode Halaman *Edit Supreme*

Halaman ini berfungsi untuk mengedit data *supreme* (*agency*) yang spesifik, disajikan dalam formulir ringkas dan terpusat menggunakan tailwind CSS. Pengguna dapat memperbarui tiga field: ID supreme, nama *supreme*, dan skema *benefit* yang digunakan (dipilih dari dropdown skema yang tersedia). Form ini menggunakan metode PUT untuk mengirimkan pembaruan data dan dilengkapi dengan validasi error di bawah setiap field, serta tombol *update* untuk menyimpan perubahan dan tombol Back untuk navigasi kembali ke daftar *supreme*.

5.2.2.18 Halaman *Create Supreme*

Potongan kode implementasi views halaman *create supreme* dapat dilihat pada Gambar 5.2.2.18.



Gambar 5.2.2.18 Potongan Kode Halaman *Create Supreme*

Halaman ini menyediakan formulir ringkas untuk menambah *supreme* (*agency*) baru, berfokus pada tiga *field* data penting: ID *supreme*, nama *supreme*, dan skema *benefit* yang akan diterapkan (dipilih dari dropdown skema yang tersedia). Form ini menggunakan metode POST untuk menyimpan data baru dan dilengkapi dengan error message di bawah setiap *field* untuk validasi, serta tombol *save* dan tombol *back* untuk navigasi.

5.2.3 Implementasi *Controller*

Pada bagian ini dibahas mengenai implementasi controller dari aplikasi web *Sales Agent*, yang berfungsi sebagai penghubung antara *model* dan *view* dalam arsitektur MVC. Controller bertugas mengelola alur data,

memproses input pengguna, serta mengatur logika bisnis seperti penyimpanan, pembaruan, dan pengambilan data dari basis data. Berikut merupakan controller terdapat pada aplikasi web *sales agent.*

5.2.3.1. *Sales Agent Controller*

Potongan kode implementasi *controller sales agent* dapat dilihat pada Gambar 5.2.3.1.

```
12  class SalesAgentController extends Controller
122     public function create()
123     {
124         $witels = Witel::all();
125         $supremes = Supreme::all();
126
127         return view('sales_agents.create', [
128             'witels' => $witels,
129             'supremes' => $supremes,
130             'title' => 'Tambah Sales Agent',
131             'sub_title' => 'Form Tambah Sales Agent',
132         ]);
133     }
134
135     public function store(Request $request)
136     {
137         $validated = $request->validate([
138             'id' => 'required|string|unique:sales_agents,id',
139             'name' => 'required|string',
140             'witel_id' => 'required|exists:witels,id',
141             'telda_id' => 'nullable|exists:teldas,id',
142             'supreme_id' => 'required|exists:supremes,id',
143             'email' => 'required|email|unique:sales_agents,email',
144             'phone' => 'nullable|string'
```

Gambar 5.2.3.1 Potongan Kode *Sales Agent Controller*

Controller sales agent adalah *controller* yang menangani seluruh operasi *CRUD* pada *sales agent*. Fungsi *index* menampilkan daftar *Sales Agent* dengan filter dan pencarian, sementara *show* menampilkan detail seorang agent beserta relasinya. *Create* dan *store* digunakan untuk menambahkan agent baru, termasuk validasi data dan unggahan foto, sedangkan edit dan update mengelola perubahan data dan

penggantian foto. *Destroy* menghapus agent beserta fotonya dari penyimpanan. Secara keseluruhan, controller ini memastikan pengelolaan data *sales agent* lengkap, termasuk filter, relasi, dan manajemen file.

5.2.3.2. *Order Controller*

Potongan kode implementasi *controller order* dapat dilihat pada Gambar 5.2.3.2.

```php
class OrderController extends Controller
{
    1 reference | 0 overrides
    public function index(Request $request): Factory|View
    {
        $selectedWitel   = $request->witel_id;
        $selectedTelda   = $request->telda_id;
        $selectedSupreme = $request->supreme_id;
        $search          = $request->search;      // search sales (id / nama)
        $orderSearch     = $request->order_search; // 🔍 search order_id

        $query = Order::with(relations: ['witel', 'telda', 'salesAgent.witel', 'salesAgent.telda'])
            ->orderBy(column: 'order_id', direction: 'asc');

        // filter berdasarkan witel sales agent
        if ($selectedWitel) {
            $query->whereHas(relation: 'salesAgent', callback: function ($q) use ($selectedWitel): void {
                $q->where('witel_id', $selectedWitel);
            });
        }
        // filter berdasarkan telda sales agent
        if ($selectedTelda) {
            $query->whereHas(relation: 'salesAgent', callback: function ($q) use ($selectedTelda): void {
                $q->where('telda_id', $selectedTelda);
            });
        }
        // filter berdasarkan supreme multiple
        if (is_array(value: $selectedSupreme) && count(value: $selectedSupreme) > 0) {
            $query->whereHas(relation: 'salesAgent', callback: function ($q) use ($selectedSupreme): void {
                $q->whereIn('supreme_id', $selectedSupreme);
            });
        }

        // 🔍 search sales by kode_sa atau nama
        if ($search) {
            $query->whereHas(relation: 'salesAgent', callback: function ($q) use ($search): void {
                $q->where('id', 'like', "%$search%")
                  ->orWhere('name', 'like', "%$search%");
```

Gambar 5.2.3.2 Potongan Kode *Order Controller*

Controller Order ini berfungsi utama untuk mengelola tampilan dan pembaruan data Orders. Fungsi index bertugas menampilkan daftar Orders lengkap dengan eager loading relasi dan menerapkan sistem filter data yang kompleks berdasarkan Witel,

Telda, Supreme, dan pencarian langsung berdasarkan ID Order atau Kode/Nama Sales Agent. Fungsi edit menyiapkan formulir untuk mengubah data Order (khususnya Sales Agent yang terasosiasi), dan fungsi update menangani pembaruan data Order tersebut setelah memvalidasi input sales_agent_id. Controller ini secara keseluruhan berfokus pada penyediaan tampilan daftar Order yang dapat difilter dan fungsionalitas untuk mengaitkan Order dengan Sales Agent yang tepat.

5.2.3.3. *Leaderboard Controller*

Potongan kode implementasi *controller leaderboard* dapat dilihat pada Gambar 5.2.3.3.

```
13  class LeaderboardController extends Controller
15      public function index(Request $request)
51          $agents = SalesAgent::with('supreme')
52              ->when($witelId, fn($q) => $q->where('witel_id', $witelId))
53              ->when($teldaId, fn($q) => $q->where('telda_id', $teldaId))
54              ->when($supremeId, function ($q) use ($supremeId) {
55                  if (is_array($supremeId)) {
56                      return $q->whereIn('supreme_id', $supremeId);
57                  }
58                  return $q->where('supreme_id', $supremeId);
59              })
60
61              ->when($jenis && is_array($jenis), fn($q) => $q->whereIn('jenis', $jenis))
62              ->when($search, function ($q) use ($search) {
63                  $q->where(function ($query) use ($search) {
64                      $search = strtolower($search);
65                      $query->orWhereRaw('LOWER(name) LIKE ?', ["%$search%"])
66                          ->orWhereRaw('LOWER(id) LIKE ?', ["%$search%"]);
67                  });
68              })
69              ->get()
70              ->keyBy(fn($agent) => "{$agent->witel_id}-{$agent->telda_id}-{$agent->id}");
```

Gambar 5.2.3.3 Potongan Kode *Sales Agent Controller*

Leaderboard Controller berfungsi untuk menampilkan dan mengelola data peringkat penjualan *sales agent* berdasarkan periode waktu, wilayah, dan kategori tertentu. Melalui fungsi

index, controller ini memproses filter seperti tanggal, witel, telda, produk, dan jenis *sales agent* untuk menghitung total penjualan, jumlah transaksi, serta menyusun peringkat berdasarkan hasil penjualan. Fungsi detail menampilkan rincian order dari setiap *sales agent* sesuai filter yang dipilih, termasuk data order yang tidak memiliki *sales* terkait. Secara keseluruhan, controller ini mendukung analisis performa dan evaluasi kinerja penjualan secara dinamis dan terstruktur.

5.2.3.4. *Gamification Controller*

Potongan kode implementasi *controller gamification* dapat dilihat pada Gambar 5.2.3.4.

```php
class GamificationController extends Controller
{

    1 reference | 0 overrides
    public function index(Request $request): mixed
    {
        $witelId   = $request->get(key: 'witel_id');
        $teldaId   = $request->get(key: 'telda_id');
        $supremeId = $request->get(key: 'supreme_id', default: []);
        $jenis     = $request->get(key: 'jenis', default: []);

        $criteriaHsi = SalesCriteria::where(column: 'nama', operator: 'HSI')->first();
        $criteriaWms = SalesCriteria::where(column: 'nama', operator: 'WMS')->first();
        $criteriaNetmonk = SalesCriteria::where(column: 'nama', operator: 'NETMONK')->first();
        $criteriaOca = SalesCriteria::where(column: 'nama', operator: 'OCA')->first();
        $criteriaPijar = SalesCriteria::where(column: 'nama', operator: 'PIJAR SEKOLAH')->first();
        $criteriaEazy = SalesCriteria::where(column: 'nama', operator: 'Eazy')->first();


        $query = SalesAgent::with(relations: ['orders', 'sales_agent_targets']);

        if ($request->filled(key: 'witel_id')) {
            $query->where(column: 'witel_id', operator: $witelId);
        }

        if ($request->filled(key: 'telda_id')) {
            $query->where(column: 'telda_id', operator: $teldaId);
        }

        if ($request->filled(key: 'supreme_id') && is_array(value: $supremeId)) {
            $query->whereIn(column: 'supreme_id', values: $supremeId);
        }

        if ($request->filled(key: 'jenis') && is_array(value: $jenis)) {
            $query->whereIn(column: 'jenis', values: $jenis);
        }
```

Gambar 5.2.3.4 Potongan Kode *Gamification Controller*

Controller *Gamification* mengelola tampilan *leaderboard* performa sales agent. Fungsi *index* memuat data *sales agent*, menerapkan filter ekstensif (Witel, Telda, *Supreme*, dll.), menghitung metrik performa (Target, *Real*, *Score*, *Growth*) untuk 6 kriteria produk (HSI hingga Eazy), menentukan Ranking, dan menghitung total *benefit*. Fungsi *detail* menyediakan drill-down ke daftar *orders* spesifik seorang *agent* berdasarkan kriteria performa tertentu. *Controller* ini intinya adalah mesin komputasi dan pemfilteran untuk sistem gamifikasi.

5.2.3.5. *Sales Agent* Target *Controller*

Potongan kode implementasi *controller sales agent* target dapat dilihat pada Gambar 5.2.3.5.

```php
class SalesAgentTargetController extends Controller
{
    public function index(Request $request)
    {
        $productsList = ['HSI', 'WMS', 'Netmonk', 'OCA', 'Pijar', 'Eazy'];

        $bulan  = $request->input('bulan', date('n'));
        $status = $request->input('status', null);
        $search = $request->input('search', null);
        $tahun  = 2025;

        $periode = Carbon::createFromDate($tahun, $bulan, 1)->startOfMonth();

        // base query
        $query = SalesAgent::with(['sales_agent_targets' => function ($q) use ($periode)
            $q->where('periode', $periode);
        }]);

        // filter status
        if ($status !== null && $status !== '') {
            $query->where('is_active', $status);
        }

        // filter search
```

Gambar 5.2.3.5 Potongan Kode *Sales Agent* Target *Controller*

Sales Agent Target *Controller* berfungsi untuk mengelola target penjualan *sales agent*

berdasarkan produk dan periode tertentu. Melalui fungsi *index*, controller ini menampilkan daftar *sales agent* beserta target bulanan yang dapat difilter berdasarkan status dan pencarian. Fungsi *updateMultiple* memungkinkan pembaruan target secara massal, sedangkan *copyTargets* digunakan untuk menyalin target dari satu bulan ke bulan lain. Secara keseluruhan, *controller* ini mempermudah pengaturan, pembaruan, dan penyalinan target penjualan secara efisien.

5.2.3.6. *Sales Agent Benefit Controller*

Potongan kode implementasi *controller sales agent* benefit dapat dilihat pada Gambar 5.2.3.6.

Gambar 5.2.3.6 Potongan Kode *Sales Agent* Benefit *Controller*

Controller SalesAgentBenefitController mengelola seluruh operasi CRUD untuk Skema Benefit *Sales Agent*. Fungsi *index* menampilkan skema yang dikelompokkan, *store* menyimpan skema baru (matrik kuantitas vs. *benefit*), sementara edit, *update*, dan *destroy* mengelola modifikasi atau penghapusan satu rule benefit spesifik.

5.2.3.7. *Sales Agent* Kriteria *Controller*

Potongan kode implementasi *controller sales agent* kriteria dapat dilihat pada Gambar 5.2.3.7.

```
class SalesCriteriaController extends Controller
    public function index()
    {
        $salesCriteria = SalesCriteria::all()->map(function ($item) {
            // Convert dari 0-1 ke 0-100 untuk tampilan
            $item->bobot = $item->bobot * 100;
            $item->score_max = $item->score_max * 100;
            return $item;
        });

        return view('sales-criteria.index', [
            'title' => 'Pengaturan Sales Criteria',
            'sub_title' => 'Atur bobot dan score maksimal kriteria penilaian'
            'salesCriteria' => $salesCriteria
        ]);
    }


    public function update(Request $request)
    {
        foreach ($request->input('sales_criteria', []) as $id => $data) {
            SalesCriteria::where('id', $id)->update([
                // Convert dari 0-100 ke 0-1 sebelum simpan
```

Gambar 5.2.3.7 Potongan Kode *Sales Agent* Kriteria *Controller*

Sales Agent Kriteria *Controller* digunakan untuk mengelola kriteria penilaian Sales Agent seperti bobot dan skor maksimum. Fungsi *index* menampilkan

semua data kriteria dengan mengubah nilai bobot dan skor dari skala 0–1 menjadi 0–100 agar lebih mudah dibaca di tampilan. Fungsi update memperbarui data tersebut berdasarkan input pengguna, lalu mengonversi kembali nilainya ke skala 0–1 sebelum disimpan ke database. Secara singkat, controller ini memfasilitasi penyesuaian bobot dan skor penilaian sales secara efisien.

5.2.3.8. *Supreme Controller*

Potongan kode implementasi *controller supreme* dapat dilihat pada Gambar 5.2.3.8.

```php
class SupremeController extends Controller
{
    /**
     * Tampilkan semua Supreme beserta skema benefit-nya.
     */
    0 references | 0 overrides
    public function index(): Factory|View
{
    $supremes = Supreme::with(relations: 'salesAgentBenefits')
                        ->orderBy(column: 'id', direction: 'asc') // urut sesuai id ascending
                        ->get();

    $title = "Daftar Supreme";
    $sub_title = "Supreme";
    return view(view: 'supremes.index', data: compact(var_name: 'supremes', var_names: 'title', 'sub_title'));
}


    /**
     * Tampilkan form untuk membuat Supreme baru.
     */
    0 references | 0 overrides
    public function create(): Factory|View
    {
        $title = "Tambah Supreme";
        $sub_title = "Supreme";

        // Ambil semua skema unik dari tabel SalesAgentBenefit
        $skemas = SalesAgentBenefit::select('id_skema')
            ->distinct()
            ->orderBy('id_skema', 'asc')
            ->get();

        return view(view: 'supremes.create', data: compact(var_name: 'title', var_names: 'sub_title', 'skemas'));
    }
```

Gambar 5.2.3.8 Potongan Kode *Supreme Controller*

Controller SupremeController menangani seluruh operasi CRUD untuk data *Supreme* (Agency).

Fungsi *index* menampilkan semua *Supreme*, termasuk *detail* SalesAgentBenefits mereka, diurutkan berdasarkan ID. Fungsi *create* dan *store* mengelola penambahan Supreme baru dengan mewajibkan ID unik, nama, dan mengaitkannya dengan skema *benefit* yang dipilih. Terakhir, edit dan *update* memproses perubahan data *supreme*, termasuk skema *benefit*, sementara *destroy* bertanggung jawab menghapus data *supreme* yang dipilih.

5.2.4 Implementasi *Routes*

Pada bagian ini membahas implementasi routes pada aplikasi *sales agent*, yang mengatur alur navigasi antara pengguna dan sistem dengan mengarahkan setiap permintaan ke *controller* yang sesuai. Routes didefinisikan pada file web.php menggunakan *framework* Laravel untuk mengelola fitur seperti *sales agent*, *sales criteria*, dan *sales target*. Berikut adalah implementasi routes pada aplikasi *sales agent*.

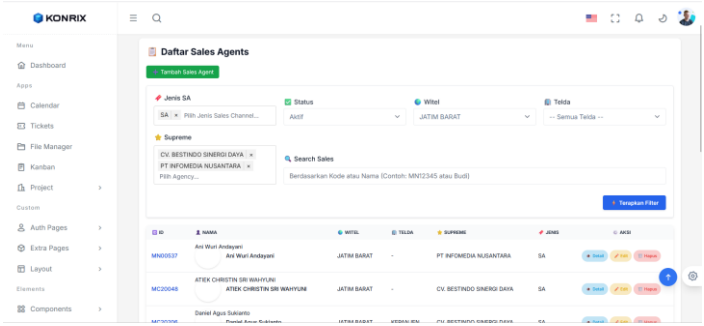
Gambar 5.2.4. Potongan Kode *Routes* Aplikasi Web *Sales Agent*

Kode ini mendefinisikan rute web yang mengatur navigasi dan akses fitur setelah login, mengarahkan ke controller untuk manajemen sales, orders, leaderboard, target, gamifikasi, dan beberapa endpoint API, semuanya dilindungi middleware auth.

5.3 Tampilan Antarmuka

Berdasarkan implementasi yang telah diterapkan dalam *views* (*front-end*), maka didapatkan hasil akhir berupa sebuah aplikasi web manajemen *sales agent* dengan gamifikasi dan *leaderboard* berbasis laravel dengan tampilan akhir sebagai berikut.

5.3.1 Tampilan Antarmuka Halaman *Index Sales Agent*

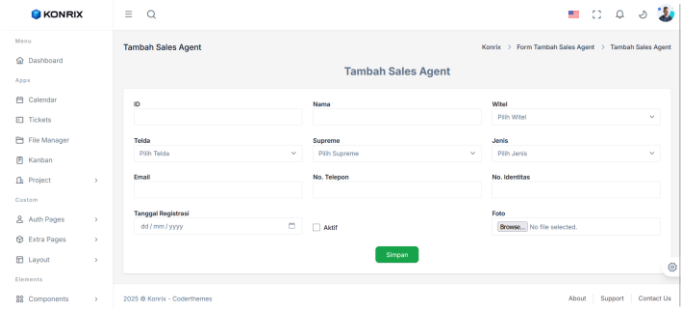


Gambar 5.3.1 Tampilan Antarmuka Halaman *Index Sales Agent*

Tampilan antarmuka halaman *index sales agent* menunjukkan daftar *sales agent* yang bisa difilter sesuai

jenis, status, witel, telda, supreme, dan bisa dicari berdasarkan nama atau kode *sales*.

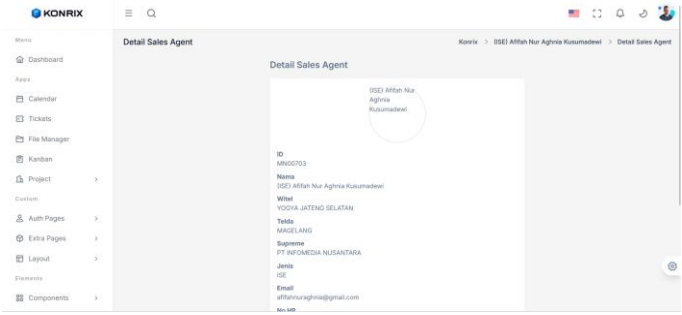
5.3.2 Tampilan Antarmuka Halaman *Create New Sales Agent*



Gambar 5.3.2 Tampilan Antarmuka Halaman *Create New Sales Agent*

Tampilan antarmuka halaman *create new sales agent* menampilkan form untuk menambahkan data sales agent baru.

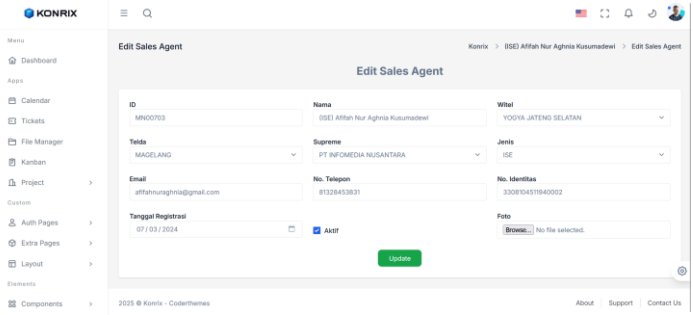
5.3.3 Tampilan Antarmuka Halaman *Show Sales Agent*



Gambar 5.3.3 Tampilan Antarmuka Halaman *Create New Sales Agent*

Tampilan antarmuka halaman *show sales agent* menampilkan detail informasi *sales agent*.

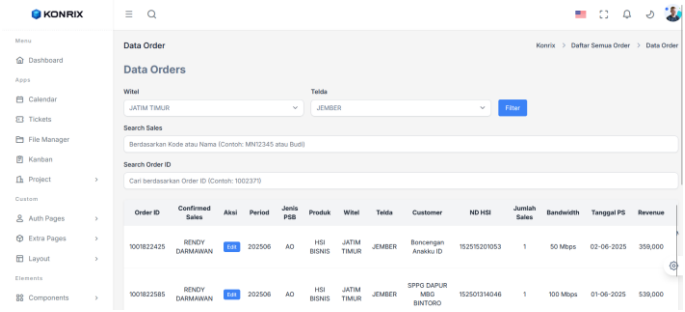
5.3.4 Tampilan Antarmuka Halaman Edit *Sales Agent*



Gambar 5.3.4 Tampilan Antarmuka Halaman *Create New Sales Agent*

Tampilan antarmuka halaman edit *sales agent* menampilkan form untuk mengedit data *sales agent*.

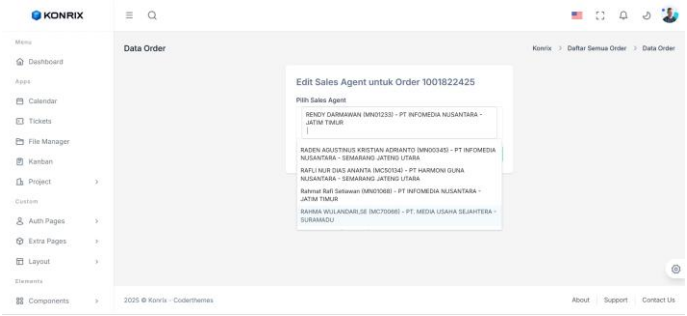
5.3.5 Tampilan Antarmuka Halaman *Index Orders*



Gambar 5.3.5 Tampilan Antarmuka Halaman *Index Orders*

Tampilan antarmuka halaman *index orders* menampilkan daftar penjualan *sales agent* yang bisa difilter berdasarkan witel dan telda *sales*.

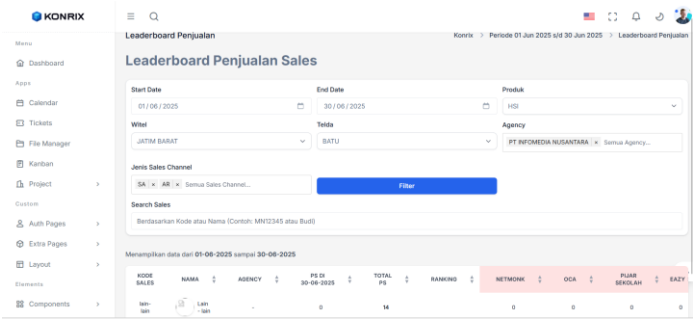
5.3.6 Tampilan Antarmuka Halaman Edit *Orders*



Gambar 5.3.6 Tampilan Antarmuka Halaman Edit *Orders*

Tampilan antarmuka halaman edit *orders* menampilkan form untuk mengubah kepemilikan penjualan (*sales agent*).

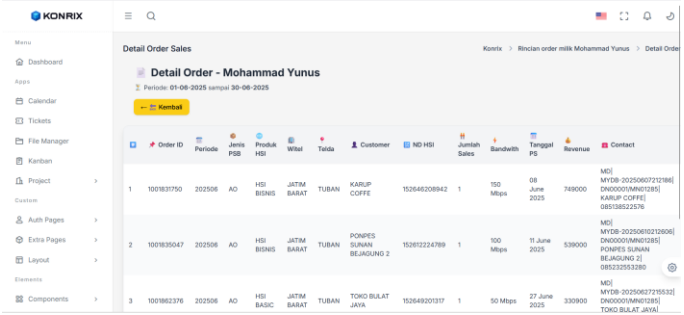
5.3.7 Tampilan Antarmuka Halaman *Index Leaderboard*



Gambar 5.3.7 Tampilan Antarmuka Halaman *Index Leaderboard*

Tampilan antarmuka halaman *index leaderboard* menampilkan *leaderboard* penjualan *sales agent*.

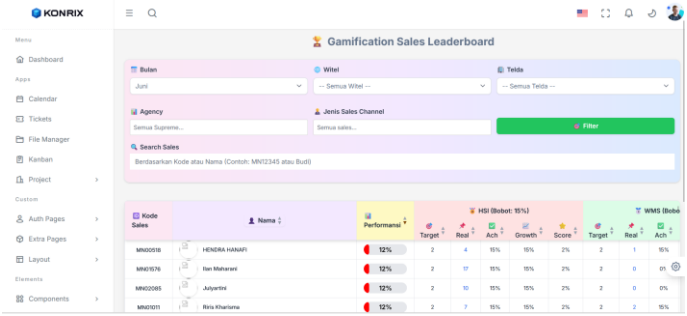
5.3.8 Tampilan Halaman Detail *Leaderboard*



Gambar 5.3.8 Tampilan Antarmuka Halaman Detail *Leaderboard*

Tampilan antarmuka halaman detail *leaderboard* menampilkan informasi detail mengenai *order* atau penjualan *sales agent*.

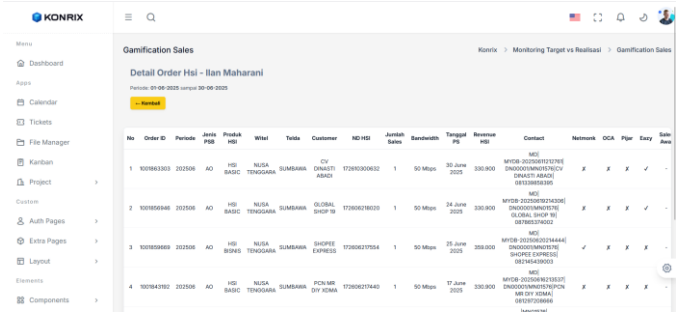
5.3.9 Tampilan Halaman *Index Gamification*



Gambar 5.3.9 Tampilan Antarmuka Halaman *Index Gamification*

Tampilan antarmuka halaman *index gamification* menampilkan performansi, target, *real, achieve, growth,* dan *score sales agent.*

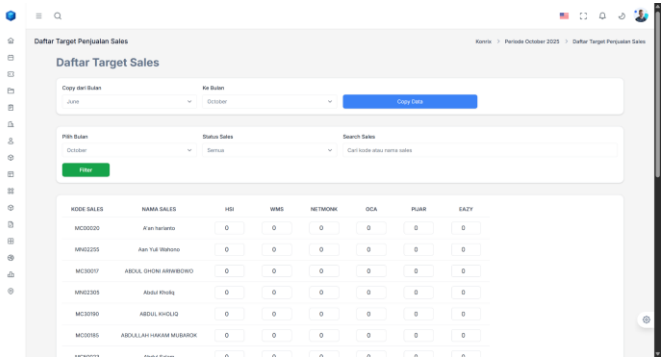
5.3.10 Tampilan Halaman Detail *Gamification*



Gambar 5.3.10 Tampilan Antarmuka Halaman Detail *Gamification*

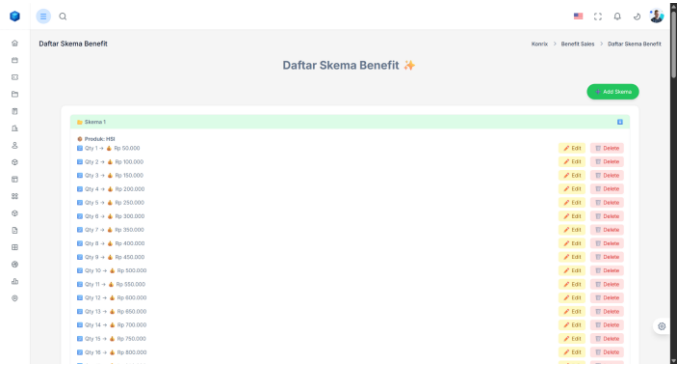
Tampilan antarmuka halaman *detail gamification* menampilkan detail order penjualan sales dalam gamifikasi.

5.3.11 Tampilan Halaman Index Sales Agent Target



Gambar 5.3.11 Tampilan Antarmuka Halaman Detail *Index Sales Agent Target*

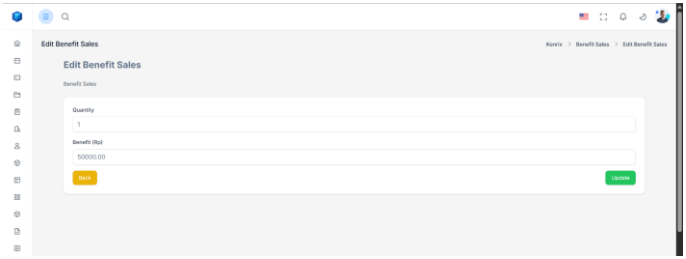
5.3.12 Tampilan Halaman Index Sales Agent Benefit



Gambar 5.3.12 Tampilan Halaman *Index Sales Agent Benefit*

Tampilan antarmuka halaman *index sales agent benefit.* Halaman ini menampilkan antarmuka pengelolaan daftar skema benefit untuk Sales Agent. Terlihat satu skema (skema 1) yang detailnya menunjukkan *benefit* (dalam Rupiah) yang akan diterima berdasarkan kuantitas penjualan produk digital. Setiap baris *qty*/*benefit* dilengkapi dengan tombol *edit* dan *delete*, serta terdapat tombol *add* skema untuk membuat skema benefit baru.

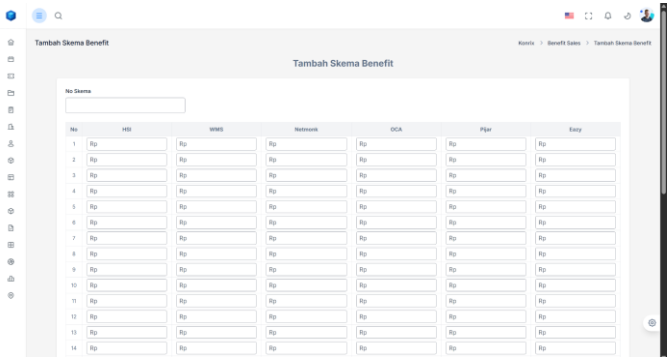
5.3.13 Tampilan Halaman *Edit Sales Agent Benefit*



Gambar 5.3.13 Tampilan Halaman *Edit Sales Agent Benefit*

Antarmuka ini adalah halaman edit *benefit sales* untuk mengubah pasangan nilai kuantitas dan nominal benefit Rp dengan tombol *update* dan *back*.

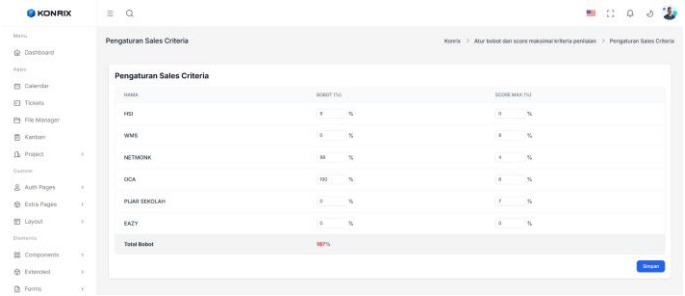
5.3.14 Tampilan Halaman Create Sales Agent Benefit



Gambar 5.3.14 Tampilan Halaman *Create Sales Agent Benefit*

Antarmuka ini digunakan untuk membuat skema *benefit* baru. Pengguna harus mengisi kolom no skema sebagai identitas skema. Di bawahnya, terdapat tabel yang memungkinkan pengguna memasukkan nilai *benefit* dalam Rupiah (Rp) untuk setiap tingkatan.

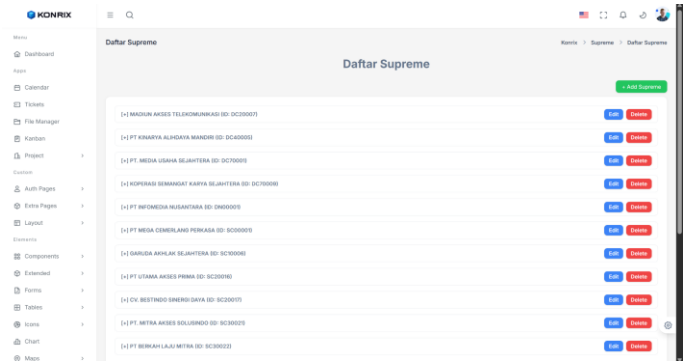
5.3.15 Tampilan Halaman *Index Sales Agent* Kriteria



Gambar 5.3.15 Tampilan Halaman *Index Sales Agent* Kriteria

Halaman ini adalah antarmuka pengaturan *sales criteria* yang digunakan untuk menentukan bobot (%) dan skor maksimal (%) untuk berbagai kriteria penjualan (seperti HSI, WMS, NETMONK, dll.). Terdapat kolom input untuk mengatur persentase bobot dan skor maksimal untuk setiap kriteria, dan di bagian bawah ditampilkan total bobot.

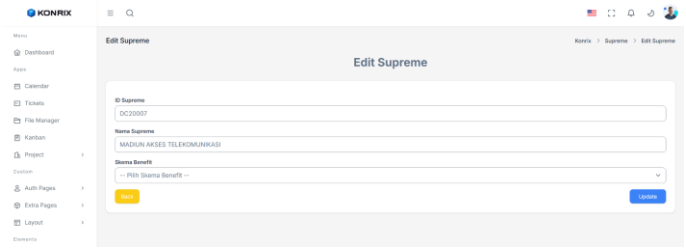
5.3.16 Tampilan Halaman *Index Supreme*



Gambar 5.3.16 Tampilan Halaman *Index Supreme*

Halaman ini menampilkan daftar entitas *supreme*, masing-masing teridentifikasi dengan nama perusahaan dan ID unik. Antarmuka ini memungkinkan pengguna untuk mengelola daftar tersebut dengan tombol *add supreme* di kanan atas, serta tombol *edit* dan *delete* di samping setiap entitas yang terdaftar.

5.3.17 Tampilan Halaman *Edit Supreme*



Gambar 5.3.17 Tampilan Halaman *Edit Supreme*

Halaman ini merupakan antarmuka edit *supreme* yang digunakan untuk memodifikasi detail entitas *supreme* yang sudah ada.

5.3.18 Tampilan Halaman *Create Supreme*

Gambar 5.3.18 Tampilan Halaman *Create Supreme*

Tampilan antarmuka halaman tambah *supreme* baru menyediakan formulir untuk input data *supreme* baru. Input yang diperlukan meliputi ID *supreme*, nama *supreme*, dan pilihan skema *benefit*.

[Halaman ini sengaja dikosongkan]

BAB VI
PENGUJIAN DAN EVALUASI

Bab ini menjelaskan tahap uji coba dilakukan terhadap aplikasi web manajemen *sales agent* dengan gamifikasi dan *leaderboard* berbasis laravel. Pengujian dilakukan untuk memastikan kualitas perangkat lunak yang dibangun dan kesesuaian hasil eksekusi perangkat lunak dengan analisis dan perancangan perangkat lunak.

6.1. Tujuan Pengujian

Pengujian dilakukan terhadap aplikasi web manajemen *sales agent* dengan gamifikasi dan *leaderboard* berbasis laravel guna menguji kesesuaian dan ketepatan fungsionalitas dari seluruh sistem aplikasi dengan acuan website tomcat.

6.2. Kriteria Pengujian

Penilaian atas pencapaian tujuan pengujian didapatkan dengan memperhatikan beberapa hasil yang diharapkan berikut :

a. Kemampuan Menampilkan *Form*

- *Form* Tambah/Edit Data : Verifikasi form untuk menambahkan atau mengedit data *sales agent*, order, dan target penjualan dapat ditampilkan dengan lengkap dan memvalidasi input pengguna dengan benar.

b. Kemampuan Menampilkan Data

- Data *Sales Agent* : Verifikasi bahwa aplikasi menampilkan daftar lengkap *sales agent* beserta atribut seperti nama, witel, telda, agency, status, dan jenis.
- Data *Order* **:** Verifikasi bahwa aplikasi dapat menampilkan data order dengan informasi lengkap, serta menampilkan relasi antara order dan sales agent.

- Leaderboard dan Gamifikasi : Uji tampilan leaderboard yang menampilkan ranking, skor, dan performa penjualan tiap *sales agent* sesuai data yang tersimpan di sistem.

c. Kemampuan Proses Data dan Analisis

- Perhitungan Skor & Ranking : Verifikasi sistem dapat menghitung skor, performa, dan peringkat secara otomatis berdasarkan data penjualan dan bobot kriteria yang telah ditetapkan.

d. Notifikasi dan Respon Sistem

- Notifikasi Update Data : Aplikasi memberikan notifikasi atau umpan balik (misalnya pesan sukses atau error) secara real-time saat data berhasil disimpan, diperbarui, atau dihapus.

e. Kemampuan Pencarian dan Filter Data

- Pencarian Data : Pengujian dilakukan untuk memastikan sistem dapat mencari data *sales agent*, order, dan *leaderboard* berdasarkan kata kunci seperti nama, kode, atau agency.
- Filter Data : Verifikasi kemampuan sistem untuk melakukan filter berdasarkan kriteria seperti jenis, status, witel, telda, *agency*, dan produk digital.
- Pengurutan Data: Pastikan hasil leaderboard dan gamifikasi dapat diurutkan berdasarkan skor, target, pencapaian, performa, atau ranking.

f. Kesesuaian Kebutuhan Non-Fungsional

- Akses Jaringan : Aplikasi web harus dapat diakses melalui jaringan internet secara stabil dan tetap berfungsi normal di berbagai perangkat.
- Antarmuka Pengguna : Pastikan tampilan antarmuka (*user interface*) mudah digunakan, konsisten, dan responsif pada berbagai ukuran layar.
- Kecepatan Akses : Verifikasi bahwa waktu respon untuk mengakses data dari database tidak melebihi

batas wajar (kurang dari 3 detik untuk permintaan umum).

g. Konsistensi Data dan Sinkronisasi Sistem

- Konsistensi Data : Verifikasi bahwa perubahan data yang dilakukan oleh admin (seperti penambahan atau penghapusan *sales agent*) langsung tercermin pada tampilan *leaderboard* dan gamifikasi.
- Validasi Input dan Output : Uji kesesuaian hasil perhitungan performa dan ranking agar sesuai dengan logika dan bobot kriteria yang telah ditentukan.

6.3. Skenario Pengujian

Skenario pengujian dilakukan dengan melakukan peran sebagai admin ataupun *user* yang akan menjalankan fitur-fitur dan seluruh kebutuhan fungsional dari sistem. Langkah - langkah untuk setiap kebutuhan fungsional yaitu sebagai berikut :

6.3.1 Admin ataupun *User*

a. Admin ataupun *User* melihat data seluruh *sales agent*.
b. Admin ataupun *User* menambahkan *sales agent* baru.
c. Admin ataupun *User* mengedit data *sales agent*.
d. Admin ataupun *User* menghapus data *sales agent*.
e. Admin ataupun User melakukan filter *sales agent* berdasarkan jenis, status, witel, telda, *agency*, nama, dan kode.
f. Admin ataupun *User* melihat data order.
g. Admin ataupun *User* melakukan filter order berdasarkan witel, telda sales, dan order id.
h. Admin ataupun *User* mengedit kepemilikan order *sales agent*.

i. Admin ataupun *User* melihat *leaderboard* penjualan sales sesuai rentang tanggal tertentu.
j. Admin ataupun *User* melakukan filter *leaderboard* berdasarkan jenis, status, witel, telda, *agency*, nama, dan kode sales.
k. Admin ataupun *User* mengurutkan *leaderboard* berdasarkan tanggal, nama *sales agent*, *agency*, total penjualan, ranking, serta jenis produk (HSI,WMS, Netmonk, OCA, Pijar, Eazy).
l. Admin ataupun *User* melihat detail order penjualan produk di dalam *leaderboard*.
m. Admin ataupun *User* melihat performa s*ales agent* yang mencakup skor total, ranking, benefit, target, realisasi, pencapaian, pertumbuhan, dan skor tiap produk digital dalam gamifikasi.
n. Admin ataupun *User* melakukan filter gamifikasi berdasarkan bulan, witel, telda, *agency,* jenis, kode, dan nama *sales agent*.
o. Admin ataupun *User* mengurutkan gamifikasi berdasarkan nama, performa, target, realisasi, pencapaian, pertumbuhan, skor, skor total, dan ranking.
p. Admin ataupun *User* melihat detail order penjualan untuk tiap produk di dalam gamifikasi.
q. Admin ataupun *User* menambahkan target penjualan *sales agent* untuk tiap produk.
r. Admin ataupun *User* menetapkan bobot kriteria tiap produk.

6.4. Evaluasi Pengujian

Hasil pengujian dilakukan terhadap pengamatan mengenai perilaku aplikasi web manajemen *sales agent* dengan gamifikasi dan *leaderboard* terhadap kasus skenario uji coba. Tabel 6.1 di bawah ini menjelaskan hasil uji coba terhadap aplikasi yang telah dibuat.

Tabel 6.1. Hasil Evaluasi Pengujian

Kriteria Pengujian	Hasil Pengujian
Melihat data seluruh *sales agent*.	Terpenuhi
Menambahkan *sales agent* baru.	Terpenuhi
Mengedit data *sales agent*.	Terpenuhi
Menghapus data *sales agent*.	Terpenuhi
Melakukan filter *sales agent* berdasarkan jenis, status, witel, telda, agency, nama, dan kode.	Terpenuhi
Melihat data order.	Terpenuhi
Melakukan filter order berdasarkan witel, telda sales, dan order id.	Terpenuhi
Mengedit kepemilikan order *sales agent*.	Terpenuhi
Melihat leaderboard penjualan sales sesuai rentang tanggal tertentu.	Terpenuhi
Melakukan filter leaderboard berdasarkan jenis, status, witel, telda, agency, nama, dan kode sales.	Terpenuhi
Mengurutkan leaderboard berdasarkan tanggal, nama *sales*	Terpenuhi

agent, agency, total penjualan, ranking, serta jenis produk (HSI,WMS, Netmonk, OCA, Pijar, Eazy).	
Melihat detail order penjualan produk di dalam leaderboard.	Terpenuhi
Melihat performa *sales agent* yang mencakup skor total, ranking, benefit, target, realisasi, pencapaian, pertumbuhan, dan skor tiap produk digital dalam gamifikasi.	Terpenuhi
Melakukan filter gamifikasi berdasarkan bulan, witel, telda, agency, jenis, kode, dan nama *sales agent*.	Terpenuhi
Mengurutkan gamifikasi berdasarkan nama, performa, target, realisasi, pencapaian, pertumbuhan, skor, skor total, dan ranking.	Terpenuhi
Melihat detail order penjualan untuk tiap produk di dalam gamifikasi.	Terpenuhi
Menambahkan target penjualan *sales agent* untuk tiap produk.	Terpenuhi
Menetapkan bobot kriteria tiap produk.	Terpenuhi

[Halaman ini sengaja dikosongkan]

BAB VII
KESIMPULAN DAN SARAN

7.1. Kesimpulan

Berdasarkan perancangan dan implementasi aplikasi web manajemen sales agent dengan gamifikasi dan leaderboard di PT. Telkom Indonesia Regional V, dapat disimpulkan bahwa:

- Aplikasi web yang dikembangkan menggunakan framework Laravel dan arsitektur MVC (Model-View-Controller) berhasil dirancang dan diimplementasikan untuk mempermudah proses manajemen dan monitoring data sales agent secara terpusat.
- Fitur gamifikasi dan leaderboard telah berhasil diterapkan, yang bertujuan untuk meningkatkan motivasi dan produktivitas sales agent dengan menampilkan peringkat kinerja dan performa penjualan secara transparan.
- Aplikasi memenuhi semua kebutuhan fungsional yang direncanakan, termasuk operasi CRUD (Create, Read, Update, Delete) pada data sales agent, manajemen order, filter data yang kompleks, pengurutan leaderboard, dan pengaturan target serta bobot kriteria penilaian produk.
- Pengujian dan evaluasi menunjukkan bahwa semua kriteria pengujian fungsionalitas aplikasi (menampilkan form, menampilkan data, proses data dan analisis, notifikasi, pencarian dan filter data) telah Terpenuhi.

7.2. Saran

Berdasarkan analisis kebutuhan non-fungsional dan implementasi sistem Aplikasi Web Manajemen Sales Agent, berikut adalah saran pengembangan lebih lanjut:

- Peningkatan Kecepatan Akses Data dan Performa (*Performance Enhancement*)
    - Meskipun pengujian fungsionalitas telah terpenuhi, untuk memenuhi Kebutuhan Non-Fungsional F-005 , disarankan untuk secara rutin memonitor waktu respon agar tidak melebihi batas wajar (kurang dari 3 detik untuk permintaan umum).
    - Lakukan indexing pada kolom-kolom *database* yang sering digunakan sebagai parameter *filter* dan pencarian, seperti kode, nama, witel_id, dan telda_id di tabel sales_agents dan orders, untuk menaikkan performa *query*.
    - Pertimbangkan mekanisme *caching* untuk data yang bersifat statis atau jarang berubah, seperti daftar witels dan teldas, serta hasil perhitungan *leaderboard* dan gamifikasi yang membutuhkan proses komputasi berat, untuk mengurangi beban *database*.
    - Lakukan pengujian kinerja (*load testing*) secara berkala untuk memverifikasi kemampuan sistem dalam menangani banyak permintaan (*request* per detik) seiring dengan bertambahnya jumlah sales agent dan data order.
- Peningkatan Keamanan dan Integritas Sistem (*Security and Integrity*)
    - Meningkatkan mekanisme autentikasi, otorisasi, dan enkripsi untuk menjamin bahwa seluruh data dalam aplikasi terlindung dari akses yang tidak

berwenang, sesuai Kebutuhan Non-Fungsional F-006.
    - Memperluas fitur audit log (Kebutuhan Non-Fungsional F-004) untuk merekam setiap aktivitas *admin* maupun *user* secara lebih rinci, terutama pada operasi sensitif seperti penambahan, pengeditan, atau penghapusan data sales agent dan target/benefit, untuk mempermudah pelacakan dan *troubleshooting*.
    - Memastikan konsistensi data (Kebutuhan Non-Fungsional F-007) melalui validasi yang ketat dan sinkronisasi data yang dilakukan oleh sistem, misalnya perubahan status sales agent harus langsung tercermin pada tampilan *leaderboard* dan gamifikasi.
- Penyempurnaan Fitur dan *User Experience* (UX)
    - Meskipun tampilan sudah responsif dan konsisten (Kebutuhan Non-Fungsional F-002 dan F-003), dapat dilakukan tinjauan UX lebih lanjut pada halaman yang kompleks seperti Gamifikasi dan Leaderboard untuk memastikan *user* dapat dengan mudah memahami skor total, *ranking*, *benefit*, dan pertumbuhan yang ditampilkan.
    - Menambahkan fitur untuk membandingkan performa beberapa sales agent secara langsung di halaman Gamifikasi, sebagai pengembangan dari alur kejadian alternatif yang sudah direncanakan.
    - Mengembangkan *tool* atau fitur untuk membandingkan target yang sudah ada ketika *admin* ingin mengubah target atau menyalin target dari bulan sebelumnya, untuk mempermudah proses manajemen target penjualan.

[Halaman ini sengaja dikosongkan]

DAFTAR PUSTAKA

[1] Duckett, J. (2011). *HTML and CSS: Design and Build Websites.* Wiley.
[2] Flanagan, D. (2020). *JavaScript: The Definitive Guide* (7th ed.). O'Reilly Media.
[3] Hunt, A., & Schuster, S. (2020). *Refactoring UI: The Book.* Independent.
[4] Repository.dinamika.ac.id. (n.d.). *BAB II.* Retrieved from https://repository.dinamika.ac.id/id/eprint/2141/4/BAB_II.pdf
[5] Stauffer, J. (2021). *Laravel: Up & Running* (3rd ed.). O'Reilly Media.
[6] Stonebraker, M., & Rowe, L. A. (2017). *The PostgreSQL Reference Manual.* PostgreSQL Global Development Group.
[7] You, E. (2021). *Vite: Next Generation Frontend Tooling. Vite Documentation.*

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS I

Nama	: Helsa Sriprameswari Putri
Tempat, Tanggal Lahir	: Surakarta, 3 Juni 2004
Jenis Kelamin	: Perempuan
Telepon	: +6281319156659
Email	: helsasp@gmail.com

AKADEMIS

Kuliah	: Departemen Teknik Informatika – FTEIC , ITS
Angkatan	: 2022
Semester	: 7 (Tujuh)

BIODATA PENULIS II

Nama	: Aurelia Natasya Putri
Tempat, Tanggal Lahir	: Malang, 25 Agustus 2004
Jenis Kelamin	: Perempuan
Telepon	: +6285233009269
Email	: aurelnandaz03@gmail.com

AKADEMIS

Kuliah	: Departemen Teknik Informatika – FTEIC , ITS
Angkatan	: 2022
Semester	: 7 (Tujuh)