WEB TECHNOLOGY

Rachmad Andri Atmoko, S.ST, M.T

Disusun Oleh:

1. Talitha Rizqa Zayyanti (244140207111052)

2. Muhammad Hafizh Al Furqon (233140700111042)

3. Faiz Henri Kurniawan (233140700111048)

UNIVERSITAS BRAWIJAYA

MALANG

2025

DAFTAR ISI

BAB 1 PENDAHULUAN

1.1. Latar Belakang Proyek

Indonesia menghadapi tantangan serius dalam pengelolaan sampah, dengan produksi mencapai 64 juta ton per tahun, namun hanya sekitar 7-10% yang berhasil didaur ulang (KLHK, 2023). Sebagian besar sampah berakhir di TPA yang sudah kelebihan muatan, seperti Bantargebang yang beroperasi pada 120% kapasitasnya, atau lebih buruk lagi, mencemari sungai dan lautan. Ironisnya, di sisi lain, industri daur ulang nasional justru mengalami kesulitan dalam memenuhi kebutuhan pasokan bahan baku karena rantai pasok yang terfragmentasi dan tidak efisien.

Permasalahan ini berakar pada tiga isu utama:

1. **Kesenjangan Informasi:** Penghasil sampah (baik rumah tangga maupun UMKM) seringkali tidak mengetahui ke mana harus menyalurkan limbah terpilah mereka. Sebaliknya, pabrik daur ulang kesulitan memantau ketersediaan bahan baku secara real-time.
2. **Inefisiensi Logistik:** Proses pengangkutan sampah dari sumber ke fasilitas daur ulang seringkali tidak terencana dengan baik. Rute yang tidak optimal menyebabkan peningkatan biaya operasional yang signifikan mencapai Rp 1,2 juta per hari per truk di Kabupaten Bandung (DLH, 2022) dan pemborosan waktu serta bahan bakar.
3. **Partisipasi Masyarakat yang Rendah:** Kurangnya insentif dan edukasi yang efektif menyebabkan tingkat partisipasi masyarakat dalam memilah sampah masih rendah, tercatat hanya sekitar 20% yang aktif melakukannya (BPS, 2022).

Menjawab tantangan multidimensional ini, aplikasi **EcoBarter** dikembangkan sebagai solusi digital inovatif. Dengan memanfaatkan teknologi web modern dan Sistem Informasi Geografis (GIS), EcoBarter bertujuan untuk menciptakan sebuah marketplace terpusat yang efisien, transparan, dan dapat diakses oleh semua pihak, sejalan dengan tema *"Sustainable Digital Solutions: Enhancing Innovation through Web Technology"*. Platform ini dirancang untuk menjembatani kesenjangan informasi, mengoptimalkan logistik, dan mendorong partisipasi masyarakat melalui sistem insentif, guna membangun ekosistem ekonomi sirkular yang kuat di Indonesia.

1.2. Tujuan dan Sasaran Aplikasi

Tujuan utama dari pengembangan aplikasi EcoBarter adalah untuk menciptakan platform digital yang dapat memperlancar dan mengoptimalkan rantai pasok limbah daur ulang.

Sasaran spesifik dari aplikasi ini adalah:

1. **Menciptakan Marketplace Terpusat:** Menyediakan sebuah platform tunggal di mana penjual dapat dengan mudah menawarkan limbah yang telah dipilah dan pembeli (industri daur ulang/pengrajin) dapat dengan mudah menemukan bahan baku sesuai kebutuhan.
2. **Meningkatkan Efisiensi Logistik:** Mengintegrasikan fitur geospasial (GIS) untuk memvisualisasikan lokasi, mencari penjual terdekat, serta menyediakan fitur pelacakan pengiriman (live tracking) untuk meningkatkan efisiensi dan transparansi dalam proses logistik.
3. **Meningkatkan Partisipasi Publik:** Memberikan insentif berupa poin (Eco Points) kepada pengguna yang aktif bertransaksi untuk mendorong lebih banyak orang terlibat dalam ekonomi

sirkular.

4. **Menyediakan Platform yang Aman dan Terpercaya:** Membangun sistem dengan autentikasi yang aman dan alur transaksi yang jelas untuk membangun kepercayaan di antara para pengguna.

1.3. Ruang Lingkup Aplikasi

Ruang lingkup fungsionalitas aplikasi EcoBarter mencakup beberapa modul utama yang melayani empat jenis peran pengguna (Admin, Seller, Buyer, Distributor):

- **Manajemen Pengguna dan Peran:** Pengguna dapat mendaftar dengan peran spesifik, login, dan mengelola profil serta alamat mereka.
- **Manajemen Marketplace:**
    - *Seller* dapat membuat dan mengelola toko (lapak) mereka.
    - *Seller* dapat mengelola daftar produk limbah yang dijual, termasuk detail varian, stok, harga, dan gambar.
    - Pengguna dapat melakukan pencarian produk limbah dengan berbagai filter, termasuk pencarian berdasarkan lokasi terdekat.
- **Manajemen Transaksi:** Sistem menangani seluruh alur transaksi, mulai dari pemesanan oleh *Buyer*, konfirmasi oleh *Seller*, hingga pembatalan.
- **Manajemen Logistik dan GIS:**
    - *Distributor* dapat melihat dan mengambil tugas pengiriman.
    - Sistem menyediakan fitur pelacakan posisi distributor secara *live* di peta.
    - Sistem menghitung estimasi jarak dan waktu tiba menggunakan layanan eksternal.
- **Sistem Gamifikasi:** Pengguna mendapatkan *Eco Points* setelah menyelesaikan transaksi yang kemudian dapat dicatat dan dilihat riwayatnya.

Di Luar Ruang Lingkup (Untuk Pengembangan Selanjutnya):

- Integrasi dengan payment gateway untuk pembayaran online otomatis.
- Penerbitan sertifikat digital berbasis teknologi blockchain.
- Integrasi langsung dengan perangkat IoT seperti *Reverse Vending Machine* (RVM).

BAB 2 ARSITEKTUR SISTEM

2.1. Gambaran Umum Arsitektur

Aplikasi **EcoBarter** dirancang menggunakan arsitektur **Single Page Application (SPA)** yang modern dan skalabel. Arsitektur ini memisahkan secara jelas antara sisi klien (Frontend) dan sisi server (Backend), di mana keduanya berkomunikasi melalui RESTful API.

- **Backend (Laravel API):** Berperan sebagai "otak" aplikasi. Backend bertanggung jawab penuh atas semua logika bisnis, otorisasi pengguna, validasi data, interaksi dengan database (termasuk query geospasial PostGIS), dan menyediakan data dalam format JSON kepada frontend. Backend tidak menangani rendering tampilan HTML secara langsung.
- **Frontend (Single Page Application):** Bertanggung jawab atas semua hal yang dilihat dan diinteraksikan oleh pengguna. Frontend akan merender antarmuka pengguna (UI), mengelola state aplikasi, dan membuat request HTTP (API calls) ke backend untuk mengambil atau mengirim data. Arsitektur ini memungkinkan pengalaman pengguna yang sangat cepat dan responsif, karena tidak perlu memuat ulang seluruh halaman saat navigasi.
- **Database (PostgreSQL + PostGIS):** Berfungsi sebagai sistem penyimpanan data yang terpusat. Penggunaan PostGIS memungkinkan penyimpanan dan pengolahan data geospasial (seperti lokasi toko, pengguna, dan distributor) secara efisien dan akurat.
- **Layanan Eksternal (API Pihak Ketiga):** Untuk fitur-fitur spesifik seperti perhitungan rute dan estimasi waktu tempuh pada fitur live tracking, sistem mengandalkan API eksternal (OSRM), yang dipanggil oleh backend.

2.2. Komponen Utama Sistem

Sistem EcoBarter terdiri dari beberapa komponen utama yang saling berinteraksi:

Modul Autentikasi & Manajemen Peran:

- Menggunakan **Laravel Sanctum** untuk autentikasi API yang aman, baik untuk SPA (berbasis sesi/cookie) maupun aplikasi mobile (berbasis token).
- Menggunakan package **spatie/laravel-permission** untuk mengelola hak akses berdasarkan peran: Admin, Seller, Buyer, dan Distributor.

Modul Marketplace:

- **Manajemen Toko (StoreController):** Memungkinkan *Seller* untuk membuat, membaca, memperbarui, dan menghapus (CRUD) informasi toko mereka.
- **Manajemen Limbah (WasteController, WasteVariantController):** Memungkinkan *Seller* untuk mengelola produk limbah dan variannya, termasuk stok dan gambar.
- **Pencarian & Filter (WasteSearchController):** Menyediakan endpoint API yang kuat untuk mencari produk limbah dengan berbagai kriteria, termasuk pencarian berbasis

lokasi.

Modul Transaksi & Logistik:

- **Manajemen Transaksi (TransactionController):** Mengelola seluruh alur transaksi dari pemesanan (pending) hingga pengiriman selesai (delivered) atau dibatalkan (cancelled).
- **Live Tracking Logistik (LogisticsController):**
    - Menyediakan endpoint bagi *Distributor* untuk mengirimkan update lokasi GPS mereka secara periodik.
    - Memproses update lokasi dengan memanggil API OSRM untuk menghitung ulang jarak dan estimasi waktu tiba.
    - Menyediakan endpoint bagi *Buyer* dan *Seller* untuk mengambil data lokasi terakhir distributor.

Modul Geospasial (GIS):

- **Integrasi PostGIS:** Menggunakan package **clickbar/laravel-magellan** untuk mempermudah penggunaan tipe data geography(Point) dan fungsi spasial di Laravel.
- **Perhitungan Spasial:** Melakukan query seperti ST_Distance untuk mengurutkan toko/produk berdasarkan jarak terdekat.

Modul Gamifikasi:

- **Eco Points System:** Mekanisme untuk memberikan poin kepada pengguna setelah menyelesaikan transaksi yang bermanfaat bagi lingkungan, dicatat dalam tabel eco_point_logs.

2.3. Teknologi yang Digunakan

Berikut adalah rincian tumpukan teknologi (tech stack) yang digunakan untuk membangun proyek EcoBarter:

Backend:

- **Bahasa:** PHP 8.2
- **Framework:** Laravel 12
- **Database:** PostgreSQL dengan ekstensi **PostGIS**.
- **Paket Utama:**
    - laravel/sanctum: Untuk autentikasi API.
    - spatie/laravel-permission: Untuk manajemen peran & izin.
    - clickbar/laravel-magellan: Untuk integrasi PostGIS yang mudah dan modern.

Frontend:

- **Arsitektur:** Single Page Application (SPA).
- **Framework JavaScript:** [Sebutkan framework Anda, misal: Vue.js, React, atau Angular]
- **Pustaka Peta Interaktif:** [Sebutkan pustaka Anda, misal: Leaflet.js atau Mapbox GL JS]
- **Styling:** [Sebutkan framework/library CSS Anda, misal: Tailwind CSS atau Bootstrap]

Layanan Pihak Ketiga & API Eksternal:

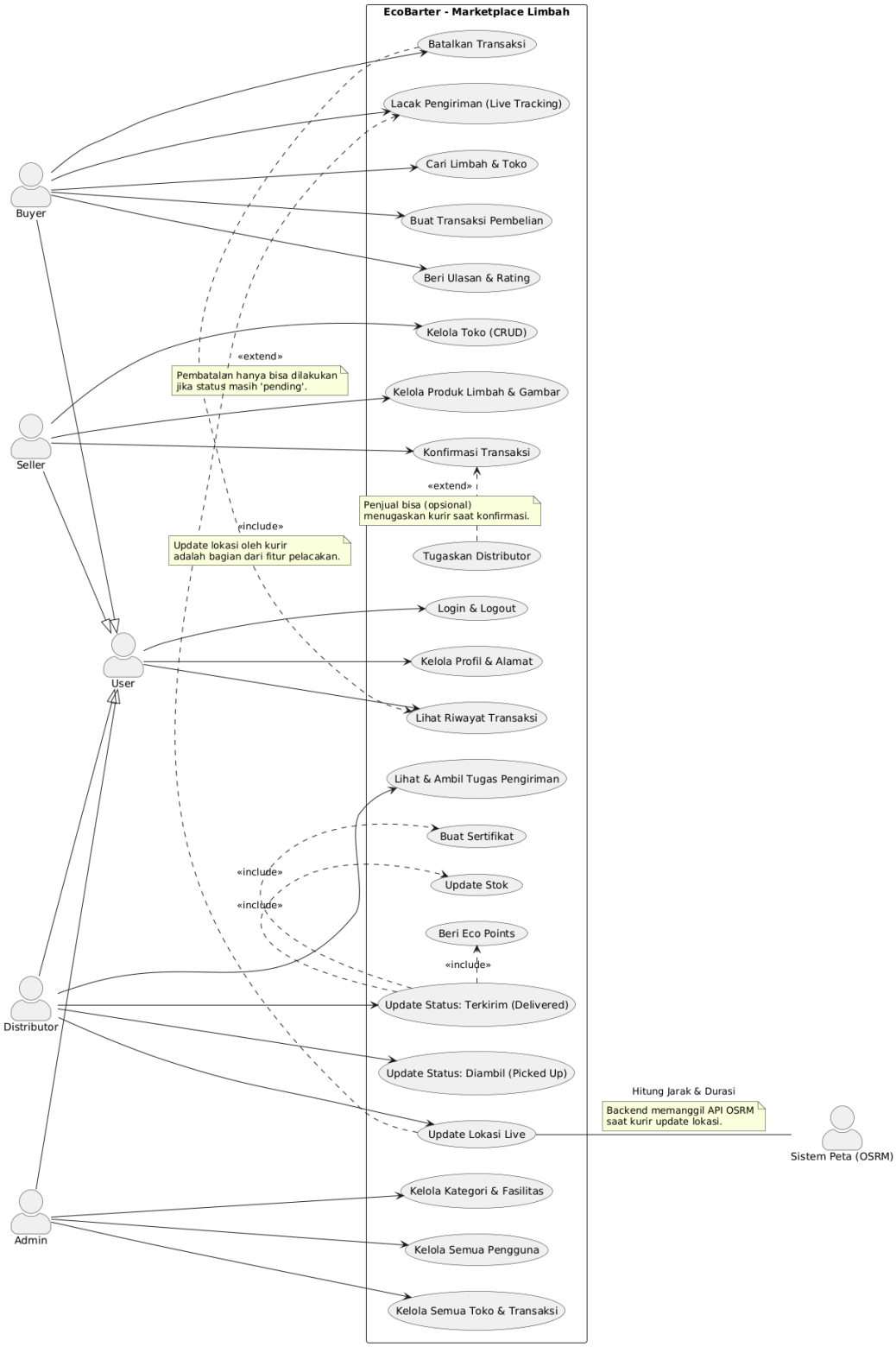
- **Peta Dasar:** OpenStreetMap (sesuai ketentuan lomba).
- **API Rute & Navigasi:** OSRM (Open Source Routing Machine) Demo Server.

Lingkungan & Deployment:

- **Server Lokal:** XAMPP.
- **Web Server:** Apache (via XAMPP).
- **Hosting:** [Sebutkan nama penyedia hosting Anda].
- **Domain & SSL:** [Sebutkan nama domain Anda dan metode penyediaan sertifikat SSL/TLS].

BAB 3 DESAIN DAN ALUR KERJA WEBSITE

3.1. Alur Kerja Utama (Use Case Diagrams/Flowcharts)

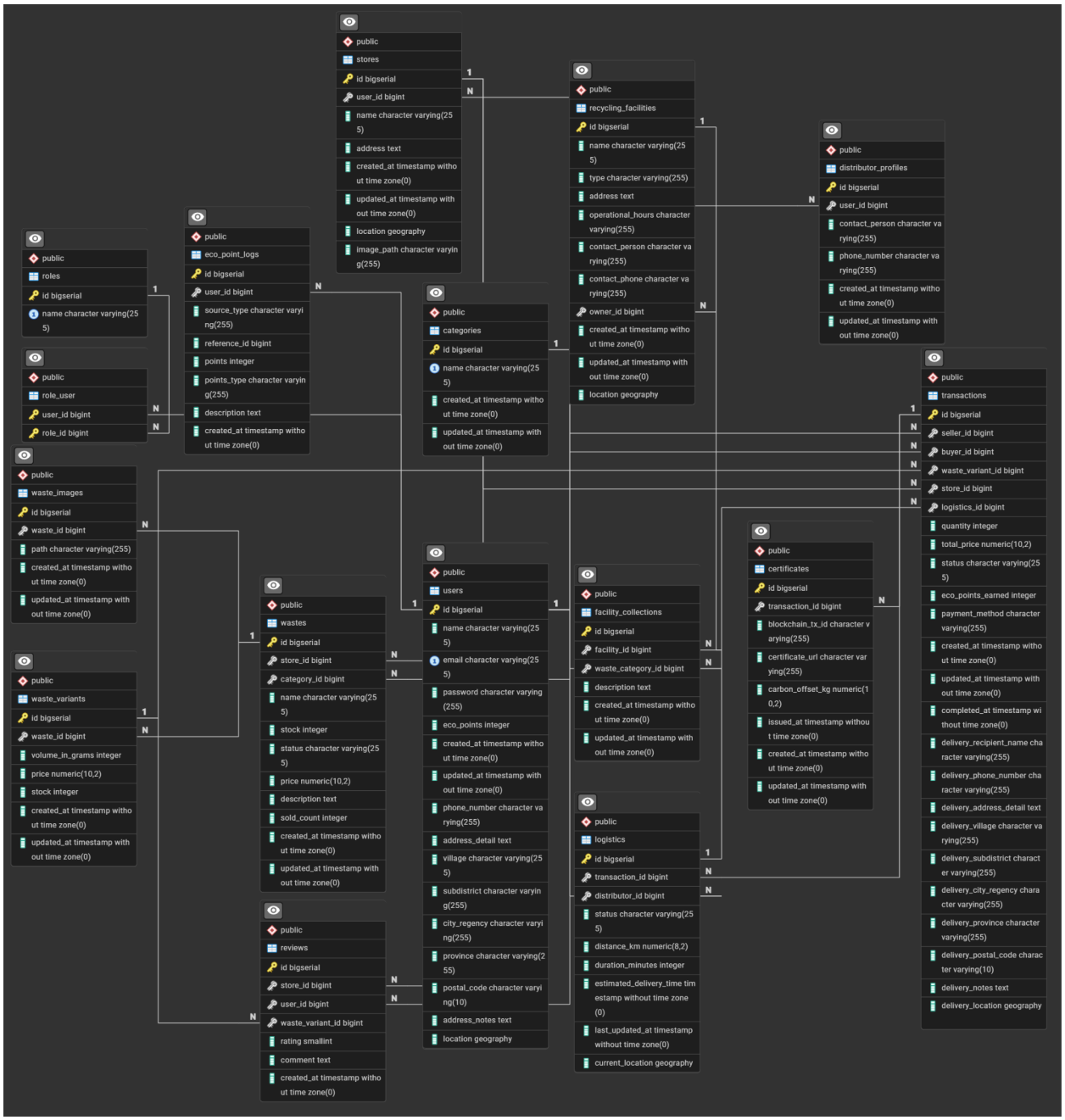


Penjelasan Diagram

1. **Aktor**:
    - **User**: Merupakan aktor dasar yang bisa login/logout dan mengelola profil.
    - **Buyer**, **Seller**, **Distributor**, **Admin**: Adalah peran spesifik yang mewarisi kemampuan User dan memiliki tugasnya masing-masing.
2. **Use Case Utama (di dalam kotak sistem):**
    - **Buyer**: Fokus pada aktivitas pembelian, mulai dari mencari barang (Cari Limbah), membuat transaksi (Buat Transaksi Pembelian), hingga melacak pengiriman (Lacak Pengiriman). Mereka juga bisa membatalkan pesanan pada kondisi tertentu (Batalkan Transaksi).
    - **Seller**: Fokus pada manajemen inventaris, seperti mengelola toko (Kelola Toko), produk limbah dan gambarnya (Kelola Produk Limbah), serta memproses pesanan yang masuk (Konfirmasi Transaksi).
    - **Distributor**: Bertanggung jawab penuh atas proses logistik, mulai dari melihat tugas yang ada (Lihat & Ambil Tugas), mengambil barang (Update Status: Picked Up), mengirimkan update lokasi untuk live tracking (Update Lokasi Live), dan menyelesaikan pengiriman (Update Status: Delivered).
    - **Admin**: Memiliki hak akses tertinggi untuk mengelola semua data master, pengguna, toko, dan transaksi di seluruh sistem.
3. **Hubungan Antar Use Case:**
    - **<<include>>**: Menunjukkan bahwa sebuah use case pasti menyertakan fungsi lain. Contoh: Saat status diubah menjadi Delivered, sistem **pasti** akan menjalankan fungsi Update Stok, Beri Eco Points, dan Buat Sertifikat.
    - **<<extend>>**: Menunjukkan fungsionalitas opsional. Contoh: Saat Konfirmasi Transaksi, seorang Seller **bisa (tapi tidak harus)** Tugaskan Distributor.
4. **Aktor Eksternal:**
    - **Sistem Peta (OSRM)**: Digambarkan sebagai aktor eksternal karena aplikasi kita berinteraksi dengannya untuk mendapatkan data rute, tetapi kita tidak mengontrol sistem tersebut.

BAB 4 IMPLEMENTASI BACKEND DAN FRONTEND (SPA)

4.1. Desain Database (Skema Relasional dan Deskripsi Tabel Utama)



Desain database aplikasi EcoBarter menggunakan PostgreSQL dengan ekstensi PostGIS untuk menangani data geospasial secara efisien. Struktur ini dirancang untuk mendukung semua fitur utama, mulai dari manajemen pengguna, marketplace, transaksi, hingga logistik dan gamifikasi.

4.1.1. Tabel users (termasuk peran dan alamat)

- **Tujuan:** Menyimpan semua data pengguna yang terdaftar di aplikasi, baik itu buyer, seller, distributor, maupun admin. Tabel ini menjadi pusat identitas dan autentikasi.
- **Kolom Kunci:**
    - id: Primary key unik untuk setiap pengguna.
    - name, email, password: Informasi kredensial dasar untuk login dan identifikasi. password disimpan dalam bentuk *hash*.
    - eco_points: Menyimpan total poin yang dimiliki pengguna, berfungsi sebagai "dompet" poin dalam sistem gamifikasi.
    - phone_number, address_detail, city_regency, dll.: Menyimpan informasi kontak dan alamat lengkap pengguna.
    - location (tipe geography(Point,4326)): Kolom PostGIS yang menyimpan koordinat **latitude dan longitude** pengguna. Ini digunakan sebagai alamat utama/default untuk pengiriman atau sebagai lokasi asal distributor.

4.1.2. Tabel roles dan role_user

- **Tujuan:** Mengimplementasikan sistem manajemen peran (*Role-Based Access Control*). Tabel ini digunakan oleh package spatie/laravel-permission untuk mengatur hak akses.
- **Kolom Kunci:**
    - roles: Menyimpan daftar peran yang ada (misalnya, 'buyer', 'seller', 'distributor', 'admin').
    - role_user: Tabel pivot yang menghubungkan users.id dengan roles.id, menandakan peran apa yang dimiliki oleh seorang pengguna. Satu pengguna bisa memiliki lebih dari satu peran.

4.1.3. Tabel stores (termasuk lokasi PostGIS)

- **Tujuan:** Menyimpan informasi mengenai "toko" atau "lapak" yang dibuat oleh pengguna dengan peran 'seller'. Setiap toko berfungsi sebagai wadah untuk produk-produk limbah yang dijual.
- **Kolom Kunci:**
    - user_id: Foreign key yang menunjukkan siapa pemilik toko ini (pengguna dengan peran 'seller').
    - name, address: Informasi dasar toko.
    - image_path: Menyimpan path ke file gambar profil toko.
    - location (tipe geography(Point,4326)): Kolom PostGIS untuk menyimpan koordinat lokasi fisik toko, memungkinkan toko ditampilkan di peta dan digunakan untuk pencarian berbasis lokasi.

4.1.4. Tabel categories, wastes, waste_variants (termasuk manajemen stok)

- **Tujuan:** Mengelola hierarki dan detail produk limbah yang dijual.
- **Kolom Kunci:**
    - categories: Tabel master untuk kategori limbah (misalnya, 'plastik', 'kertas', 'logam').
    - wastes: Merepresentasikan produk limbah utama yang dijual di sebuah toko (misalnya, "Kumpulan Botol Plastik PET"). Tabel ini terhubung ke stores dan categories. Kolom stock di sini berfungsi sebagai stok utama atau agregat, dan sold_count melacak total penjualan.
    - waste_variants: Memungkinkan satu produk limbah utama memiliki beberapa varian (misalnya, "Botol Plastik PET - 500 gram", "Botol Plastik PET - 1 kg"). Setiap varian memiliki harga dan stoknya sendiri. Logika bisnis memastikan total stock dari semua varian tidak melebihi stock di tabel wastes induknya.

4.1.5. Tabel transactions (termasuk status, metode pembayaran, alamat pengiriman snapshot)

- **Tujuan:** Merupakan tabel inti yang mencatat setiap transaksi jual-beli.
- **Kolom Kunci:**
    - seller_id, buyer_id: Mencatat siapa penjual dan pembeli dalam transaksi.
    - waste_variant_id, quantity, total_price: Mencatat item apa yang dibeli, dalam jumlah berapa, dan total harganya.
    - status: Kolom krusial yang melacak alur transaksi (pending, confirmed, picked_up, delivered, cancelled).
    - payment_method: Menyimpan metode pembayaran yang dipilih (misalnya, 'cod', 'bank_transfer_bca').
    - Kolom delivery_* (misalnya delivery_address_detail, delivery_location): Menyimpan "snapshot" atau salinan alamat pengiriman pada saat transaksi dibuat. Ini penting untuk menjaga keakuratan data historis meskipun pengguna mengubah alamat di profilnya. delivery_location bertipe PostGIS untuk keperluan logistik.

4.1.6. Tabel logistics (termasuk lokasi distributor PostGIS, status pengiriman)

- **Tujuan:** Mengelola dan melacak proses pengiriman untuk setiap transaksi.
- **Kolom Kunci:**
    - transaction_id: Menghubungkan setiap entri logistik ke satu transaksi spesifik.
    - distributor_id: Menunjukkan user_id dari kurir yang bertanggung jawab atas pengiriman ini.
    - status: Melacak status pengiriman (scheduled, in_transit, delivered, cancelled), yang berjalan paralel dengan status transaksi.
    - current_location (tipe geography(Point,4326)): Kolom PostGIS yang **diperbarui**

secara periodik oleh aplikasi distributor. Ini adalah kunci dari fitur *live tracking*.
    - distance_km, duration_minutes, estimated_delivery_time: Menyimpan data estimasi yang dihitung ulang setiap kali current_location diperbarui.

4.1.7. Tabel eco_point_logs

- **Tujuan:** Berfungsi sebagai buku catatan (ledger) untuk setiap aktivitas perolehan atau penggunaan poin. Ini memastikan semua riwayat poin transparan dan dapat diaudit.
- **Kolom Kunci:**
    - user_id: Siapa yang menerima/menggunakan poin.
    - source_type dan reference_id: Mencatat dari mana poin berasal (misalnya, source_type='transaction' dan reference_id=5 berarti poin berasal dari transaksi dengan ID 5).
    - points dan points_type: Jumlah poin dan jenisnya ('plus' atau 'minus').

4.1.8. Tabel certificates (jika diimplementasikan)

- **Tujuan:** Menyimpan referensi ke sertifikat digital yang dibuat setelah transaksi selesai, sebagai bukti otentik dari aktivitas daur ulang.
- **Kolom Kunci:**
    - transaction_id: Menghubungkan sertifikat ke transaksi yang menghasilkannya.
    - blockchain_tx_id: Menyimpan ID hash dari transaksi di jaringan blockchain, memastikan keaslian.
    - carbon_offset_kg: Menyimpan hasil perhitungan dampak lingkungan dari transaksi tersebut.

4.1.9. Tabel lain yang relevan

- **reviews**: Menyimpan ulasan dan rating yang diberikan oleh pembeli terhadap toko atau produk setelah transaksi selesai. Rating ini digunakan dalam fitur pencarian.
- **distributor_profiles**: Menyimpan informasi kontak dan detail tambahan untuk pengguna dengan peran 'distributor'.
- **recycling_facilities dan facility_collections**: Menjadi dasar untuk fitur masa depan seperti integrasi dengan RVM (Reverse Vending Machine) atau Bank Sampah, di mana pengguna bisa menyetor limbah secara langsung.

4.2. Implementasi API Endpoint Utama

4.3.1. Autentikasi

Grup Rute guest (Route::middleware('guest')->group(...))

- **Registrasi Pengguna**
  - POST /register
    - **Controller:** RegisteredUserController@store
    - **Deskripsi:** Memproses data yang dikirim dari formulir registrasi, memvalidasi input, membuat pengguna baru di database, dan secara otomatis melakukan login untuk pengguna tersebut.
- **Login Pengguna**
  - POST /login
    - **Controller:** AuthenticatedSessionController@store
    - **Deskripsi:** Memproses kredensial (email dan password) yang dikirim dari formulir login, memvalidasi, dan membuat sesi login untuk pengguna jika valid.
- **Lupa Password (Password Reset)**
  - POST /forgot-password
    - **Nama Rute:** password.email
    - **Controller:** PasswordResetLinkController@store
    - **Deskripsi:** Memvalidasi email, membuat token reset password, dan mengirim email ke pengguna yang berisi tautan untuk mereset password mereka.
  - GET /reset-password/{token}
    - **Nama Rute:** password.reset
    - **Controller:** NewPasswordController@create
    - **Deskripsi:** Menampilkan formulir untuk mengatur password baru. Rute ini diakses dari tautan yang diterima pengguna melalui email dan menyertakan token reset.
  - POST /reset-password
    - **Nama Rute:** password.store
    - **Controller:** NewPasswordController@store
    - **Deskripsi:** Memproses permintaan untuk mengatur password baru, memvalidasi token, email, dan password baru, lalu memperbarui password pengguna di database.

Grup Rute auth (Route::middleware('auth')->group(...))

Grup ini menangani semua proses yang memerlukan pengguna untuk login terlebih dahulu.

- **Verifikasi Email**
  - GET /verify-email/{id}/{hash}
    - **Nama Rute:** verification.verify

        - **Controller:** VerifyEmailController
        - **Middleware Tambahan:** signed, throttle:6,1 (melindungi dari percobaan berulang).
        - **Deskripsi:** Rute yang diakses dari tautan verifikasi di email. Rute ini akan memvalidasi tautan dan menandai email pengguna sebagai terverifikasi di database.
    - POST /email/verification-notification
        - **Nama Rute:** verification.send
        - **Controller:** EmailVerificationNotificationController@store
        - **Middleware Tambahan:** throttle:6,1.
        - **Deskripsi:** Mengirim ulang email verifikasi jika pengguna tidak menerima email pertama atau tautannya kedaluwarsa.
- **Konfirmasi Password**
    - POST /confirm-password
        - **Controller:** ConfirmablePasswordController@store
        - **Deskripsi:** Memvalidasi password yang dimasukkan pengguna.
- **Update Password**
    - PUT /password
        - **Nama Rute:** password.update
        - **Controller:** PasswordController@update
        - **Deskripsi:** Memproses permintaan untuk mengubah password pengguna dari halaman profil mereka.
- **Logout Pengguna**
    - POST /logout
        - **Nama Rute:** logout
        - **Controller:** AuthenticatedSessionController@destroy
        - **Deskripsi:** Mengakhiri sesi login pengguna saat ini dan mengarahkan mereka kembali ke halaman utama atau halaman login.

4.3.2. API untuk Stores (StoreController)

Endpoint ini digunakan untuk mengelola toko atau lapak milik para penjual.

- **GET** /api/stores

    - **Deskripsi:** Menampilkan daftar toko. Logika di backend akan menyesuaikan hasilnya berdasarkan peran pengguna (Admin melihat semua, Seller melihat tokonya, Buyer/Distributor melihat semua).
    - **Autentikasi:** Wajib.
    - **Response Sukses (JSON - 200 OK):** Mengembalikan daftar toko.
- **POST** /api/stores

    - **Deskripsi:** Membuat toko baru. Hanya bisa diakses oleh pengguna dengan peran 'Seller'.
    - **Autentikasi:** Wajib (peran 'Seller').
    - **Request Body (multipart/form-data):** name, address, latitude, longitude, dan file image (opsional).
    - **Response Sukses (JSON - 201 Created):** Mengembalikan detail toko yang baru dibuat.
- **GET** /api/stores/{store}
    - **Deskripsi:** Menampilkan detail satu toko spesifik, termasuk daftar limbah yang dijual.
    - **Autentikasi:** Wajib.
    - **Response Sukses (JSON - 200 OK):** Mengembalikan objek detail toko.
- **POST** /api/stores/{store} (dengan _method: 'PATCH')
    - **Deskripsi:** Memperbarui data toko, termasuk mengganti gambar. Hanya bisa diakses oleh pemilik toko atau admin.
    - **Autentikasi:** Wajib (pemilik/admin).
    - **Request Body (multipart/form-data):** Field yang ingin diubah (misal: name, address, latitude, longitude, image).
    - **Response Sukses (JSON - 200 OK):** Mengembalikan detail toko yang telah diperbarui.
- **DELETE** /api/stores/{store}
    - **Deskripsi:** Menghapus toko beserta semua data limbah di dalamnya. Hanya bisa diakses oleh pemilik toko atau admin.
    - **Autentikasi:** Wajib (pemilik/admin).
    - **Response Sukses (JSON - 200 OK):** Mengembalikan pesan konfirmasi penghapusan.

4.3.3. API untuk Wastes, Waste Variants, WasteSearch (WasteController, WasteVariantController, WasteSearchController)

Bagian ini mencakup endpoint untuk mengelola produk limbah (wastes), varian-variannya (waste_variants), serta endpoint publik untuk pencarian limbah (wastes/search).

WasteSearchController

Controller ini menyediakan satu endpoint publik utama yang sangat fleksibel untuk menelusuri seluruh marketplace limbah.

- **GET /api/wastes/search**
    - **Deskripsi:** Mencari produk limbah di seluruh toko dengan berbagai filter dan kriteria pengurutan. Endpoint ini dapat diakses oleh semua pengguna, termasuk yang belum login.
    - **Controller:** WasteSearchController@search
    - **Autentikasi:** Tidak Wajib.
    - **Parameter Query (Opsional):**

        - q (string): Kata kunci untuk mencari berdasarkan nama limbah (case-insensitive).
        - category (string): Nama kategori untuk memfilter hasil (contoh: kertas, plastik).
        - status (string): Memfilter berdasarkan status limbah (available, sold, expired).
        - sort_by (string): Mengurutkan hasil. Pilihan:
            - price_asc: Harga termurah.
            - rating_desc: Rating toko tertinggi.
            - sold_desc: Paling banyak terjual.
            - nearby: Terdekat dari lokasi pengguna.
        - latitude & longitude (float): Koordinat pengguna saat ini. **Wajib diisi jika sort_by=nearby**.
    - **Response Sukses (JSON - 200 OK):** Mengembalikan array objek limbah yang cocok dengan kriteria. Jika sort_by=nearby, setiap objek akan memiliki properti tambahan distance_km.

WasteController

Endpoint ini mengelola produk limbah utama (wastes) dan bersifat *nested* di dalam rute toko (stores).

- **GET /api/stores/{store}/wastes**
    - **Deskripsi:** Menampilkan daftar semua produk limbah yang dimiliki oleh sebuah toko spesifik, lengkap dengan gambar dan kategori.
    - **Controller:** WasteController@index
    - **Autentikasi:** Tidak Wajib (Publik).
    - **Response Sukses (JSON - 200 OK):** Mengembalikan data limbah yang dipaginasi.
- **POST /api/stores/{store}/wastes**
    - **Deskripsi:** Menambahkan produk limbah baru ke sebuah toko. Hanya bisa diakses oleh pemilik toko atau admin.
    - **Controller:** WasteController@store
    - **Autentikasi:** Wajib (Peran Seller pemilik toko atau Admin).
    - **Request Body (multipart/form-data):** name, category_id, stock, status, price, description, dan images[] (array file gambar, opsional, maks 5).
    - **Response Sukses (JSON - 201 Created):** Mengembalikan detail data limbah yang baru dibuat beserta URL gambarnya.
- **GET /api/wastes/{waste}**
    - **Deskripsi:** Menampilkan detail satu produk limbah spesifik.
    - **Controller:** WasteController@show
    - **Autentikasi:** Tidak Wajib (Publik).
    - **Response Sukses (JSON - 200 OK):** Mengembalikan objek detail limbah beserta gambar, kategori, dan info tokonya.
- **POST /api/wastes/{waste}** (dengan _method: 'PATCH')

    - **Deskripsi:** Memperbarui detail data limbah (misal: nama, stok, harga). Endpoint ini **tidak** untuk mengelola gambar. Penambahan/penghapusan gambar dilakukan melalui endpoint terpisah. Hanya bisa diakses oleh pemilik toko atau admin.
    - **Controller:** WasteController@update
    - **Autentikasi:** Wajib (pemilik/admin).
    - **Request Body (multipart/form-data):** Field yang ingin diubah (misal: name, stock, dll.).
    - **Response Sukses (JSON - 200 OK):** Mengembalikan detail data limbah yang telah diperbarui.
- **DELETE /api/wastes/{waste}**
    - **Deskripsi:** Menghapus data limbah beserta semua gambar dan varian yang terkait dengannya (karena onDelete('cascade')). Hanya untuk pemilik toko atau admin.
    - **Controller:** WasteController@destroy
    - **Autentikasi:** Wajib (pemilik/admin).
    - **Response Sukses (JSON - 200 OK):** Mengembalikan pesan konfirmasi penghapusan.
- **DELETE /api/waste-images/{waste_image}**
    - **Deskripsi:** Endpoint khusus untuk menghapus **satu gambar spesifik** dari sebuah produk limbah.
    - **Controller:** WasteController@destroyImage
    - **Autentikasi:** Wajib (pemilik/admin).
    - **Response Sukses (JSON - 200 OK):** Mengembalikan pesan konfirmasi.

WasteVariantController

Endpoint ini mengelola varian dari produk limbah utama dan bersifat *nested* di dalam rute limbah (wastes). Controller ini saat ini masih mengembalikan view() dan redirect(), yang perlu diubah menjadi response()->json() untuk SPA.

- **GET /api/wastes/{waste}/variants**
    - **Deskripsi:** Menampilkan semua varian dari satu produk limbah utama.
    - **Controller:** WasteVariantController@index
    - **Autentikasi:** Wajib (pemilik/admin).
    - **Response Sukses (JSON - 200 OK):** Mengembalikan daftar varian limbah yang dipaginasi.
- **POST /api/wastes/{waste}/variants**
    - **Deskripsi:** Membuat varian baru untuk sebuah produk limbah. Hanya untuk pemilik toko atau admin.
    - **Controller:** WasteVariantController@store

    - **Autentikasi:** Wajib (pemilik/admin).
    - **Request Body (JSON):** volume_in_grams, price, stock.
    - **Response Sukses (JSON - 201 Created):** Mengembalikan detail varian yang baru dibuat.
- *(Rute lain untuk show, update, destroy pada varian mengikuti pola RESTful standar dengan otorisasi yang sesuai).*

4.3.4 API untuk Transactions (TransactionController - termasuk update status)

Controller ini merupakan inti dari alur kerja jual-beli di aplikasi EcoBarter. Semua endpoint di bawah ini memerlukan autentikasi (auth:sanctum) dan diakses melalui prefix /api/transactions.

- **GET /api/transactions**
    - **Deskripsi:** Menampilkan daftar riwayat transaksi. Hasil yang ditampilkan akan difilter secara otomatis berdasarkan peran pengguna yang login: *Admin* melihat semua, *Seller* melihat penjualan, *Buyer* melihat pembelian, dan *Distributor* melihat tugas pengiriman mereka. Endpoint ini mendukung paginasi dan filter berdasarkan status.
    - **Controller:** TransactionController@index
    - **Autentikasi:** Wajib.
    - **Parameter Query (Opsional):**
        - per_page (integer): Jumlah item per halaman (default: 15).
        - status (string): Filter transaksi berdasarkan status (contoh: pending, delivered).
        - sort_direction (string): Arah pengurutan (asc untuk terlama, desc untuk terbaru).
    - **Response Sukses (JSON - 200 OK):** Mengembalikan objek paginasi yang berisi daftar transaksi.
- **POST /api/transactions**
    - **Deskripsi:** Membuat transaksi baru (memesan produk). Hanya bisa diakses oleh pengguna dengan peran 'Buyer'. Sistem akan mengambil alamat pengiriman dari profil 'Buyer' yang login dan menyimpannya sebagai snapshot di transaksi.
    - **Controller:** TransactionController@store
    - **Autentikasi:** Wajib (Peran Buyer).
    - **Request Body (JSON):** waste_variant_id, quantity, payment_method.
    - **Response Sukses (JSON - 201 Created):** Mengembalikan detail transaksi yang baru dibuat dengan status 'pending'.
    - **Response Gagal (JSON - 422 Unprocessable Entity):** Jika stok tidak mencukupi atau alamat buyer belum lengkap.
- **GET /api/transactions/{transaction}**
    - **Deskripsi:** Menampilkan detail satu transaksi spesifik. Otorisasi diterapkan untuk memastikan hanya pihak yang terlibat (buyer, seller, distributor) atau admin yang bisa melihatnya.
    - **Controller:** TransactionController@show

    - **Autentikasi:** Wajib.
    - **Response Sukses (JSON - 200 OK):** Mengembalikan objek detail transaksi beserta relasi yang dimuat (seperti detail buyer, seller, produk, dll.).
- **PATCH /api/transactions/{transaction}/confirm-by-seller**

    - **Deskripsi:** Mengubah status transaksi dari 'pending' menjadi 'confirmed'. Endpoint ini hanya bisa diakses oleh 'Seller' dari transaksi tersebut atau oleh 'Admin'. Saat konfirmasi, seller juga dapat menugaskan seorang 'Distributor'.
    - **Controller:** TransactionController@confirmBySeller
    - **Autentikasi:** Wajib (Peran Seller atau Admin).
    - **Request Body (JSON - Opsional):** { "distributor_id": ID_USER_DISTRIBUTOR }
    - **Response Sukses (JSON - 200 OK):** Mengembalikan detail transaksi yang telah diupdate.
- **PATCH /api/transactions/{transaction}/pickup-by-distributor**

    - **Deskripsi:** Mengubah status transaksi dari 'confirmed' menjadi 'picked_up'. Endpoint ini hanya untuk pengguna dengan peran 'Distributor'. Jika logistik belum dibuat, sistem akan otomatis membuatnya dan menugaskan distributor yang sedang login.
    - **Controller:** TransactionController@markAsPickedUpByDistributor
    - **Autentikasi:** Wajib (Peran Distributor).
    - **Response Sukses (JSON - 200 OK):** Mengembalikan detail transaksi yang telah diupdate.
- **PATCH /api/transactions/{transaction}/deliver-by-distributor**

    - **Deskripsi:** Mengubah status transaksi dari 'picked_up' menjadi 'delivered'. Ini adalah tahap akhir dari pengiriman oleh 'Distributor'. Setelah ini, sistem akan memfinalisasi stok, memberikan Eco Points, dan membuat sertifikat (jika ada).
    - **Controller:** TransactionController@markAsDeliveredByDistributor
    - **Autentikasi:** Wajib (Distributor yang ditugaskan).
    - **Response Sukses (JSON - 200 OK):** Mengembalikan detail transaksi yang telah selesai.
- **PATCH /api/transactions/{transaction}/cancel**

    - **Deskripsi:** Membatalkan transaksi. Aturan pembatalan bervariasi: 'Buyer' hanya bisa membatalkan saat status 'pending', sedangkan 'Seller' atau 'Admin' bisa membatalkan selama transaksi belum selesai. Sistem akan otomatis mengembalikan stok jika dibatalkan.
    - **Controller:** TransactionController@cancel
    - **Autentikasi:** Wajib.
    - **Request Body (JSON - Opsional):** { "reason": "Alasan pembatalan" }
    - **Response Sukses (JSON - 200 OK):** Mengembalikan detail transaksi yang telah dibatalkan.

4.2.6. Tabel logistics (termasuk lokasi distributor PostGIS, status pengiriman)

Controller ini merupakan inti dari fitur live tracking. Endpoint-endpoint ini dirancang untuk dipanggil oleh aplikasi distributor (untuk mengirim update lokasi) dan oleh aplikasi klien (untuk memantau pengiriman). Semua rute ini memerlukan autentikasi auth:sanctum.

- **PATCH /api/logistics/{logistics}/location**
    - **Deskripsi:** Endpoint yang digunakan secara eksklusif oleh **Distributor** yang sedang bertugas untuk mengirimkan update periodik koordinat GPS mereka. Setiap kali endpoint ini dipanggil, backend akan:
        1. Menyimpan current_location (lokasi terkini) distributor.
        2. Memanggil API eksternal (OSRM) untuk menghitung ulang jarak rute (distance_km) dan estimasi durasi (duration_minutes) ke titik tujuan.
        3. Memperbarui estimated_delivery_time berdasarkan hasil perhitungan.
        4. Menyimpan semua perubahan ke database.
    - **Controller:** LogisticsController@updateLocation
    - **Autentikasi:** Wajib. Hanya bisa diakses oleh user dengan peran 'distributor' yang distributor_id-nya cocok dengan yang tercatat di record {logistics}.
    - **Request Body (JSON):**

```
{
  "current_latitude": -7.12345,
  "current_longitude": 112.56789
}
```

Response Sukses (JSON - 200 OK):


```
{
  "message": "Lokasi dan estimasi berhasil diperbarui.",
  "logistics": {
    "id": 1,
    "transaction_id": 6,
    "distributor_id": 3,
    "status": "in_transit",
    "distance_km": "8.50",
    "duration_minutes": 15,
    "estimated_delivery_time": "2025-06-05T14:30:00.000000Z",
    "current_location": {
      "type": "Point",
      "coordinates": [112.56789, -7.12345]
    },
    "last_updated_at": "2025-06-05T14:15:00.000000Z",
    "distributor_user": {
      "id": 3,
```

```
        "name": "Budi Kurir"
      }
    }
}
```

Response Gagal (JSON):

- **403 Forbidden:** Jika pengguna bukan distributor yang ditugaskan.
- **422 Unprocessable Entity:** Jika pengiriman tidak dalam status in_transit atau jika input koordinat tidak valid.

GET /api/logistics/{logistics}/status

- **Deskripsi:** Endpoint yang digunakan oleh **Buyer** atau **Seller** (melalui SPA) untuk mengambil data logistik terkini, yang kemudian digunakan untuk menampilkan posisi distributor di peta, jarak tersisa, dan estimasi waktu tiba. Endpoint ini biasanya dipanggil secara periodik (polling).
- **Controller:** LogisticsController@getLogisticsStatus
- **Autentikasi:** Wajib. Hanya bisa diakses oleh pihak yang terlibat dalam transaksi tersebut (buyer, seller, distributor) atau oleh 'Admin'.
- **Response Sukses (JSON - 200 OK):** Mengembalikan objek detail logistik, sama seperti respons sukses dari endpoint updateLocation, beserta profil distributor.
- **Response Gagal (JSON):**
    - **403 Forbidden:** Jika pengguna yang merequest tidak terlibat dalam transaksi.
    - **404 Not Found:** Jika {logistics} ID tidak ditemukan.

4.3.5. API untuk Logistics (LogisticsController - termasuk update lokasi untuk live tracking)

BAB 5 INTEGRASI PETA (GIS)

Integrasi peta adalah fitur inti dari aplikasi EcoBarter yang memberikan konteks geografis pada data marketplace, memungkinkan pencarian berbasis lokasi, dan menyediakan fungsionalitas live tracking.

5.1. Pustaka/Layanan Peta yang Digunakan

- **Layanan Peta Dasar**: Sesuai dengan ketentuan lomba, aplikasi ini menggunakan data peta dari **OpenStreetMap (OSM)**, sebuah proyek peta dunia yang bersifat terbuka dan gratis.
- **Library Frontend**: Untuk menampilkan peta interaktif di sisi klien (browser), kami menggunakan **Leaflet.js**, sebuah pustaka JavaScript open-source yang ringan dan populer untuk menampilkan peta.
- **API Rute**: Untuk menghitung jarak rute jalan dan estimasi waktu tempuh pada fitur live tracking, kami mengintegrasikan API dari server demo publik **OSRM (Open Source Routing Machine)**.

5.2. Mekanisme Penampilan Data Geospasial

Data geospasial dari backend (yang disimpan menggunakan PostGIS) dikirim ke frontend dalam format standar **GeoJSON**. Pustaka peta di frontend kemudian mem-parsing format ini untuk menampilkan data di peta.

5.2.1. Menampilkan Lokasi Toko

1. **Pengambilan Data**: Frontend memanggil endpoint API GET /api/stores atau GET /api/wastes/search.
2. **Respons API**: Backend mengembalikan daftar toko atau limbah, di mana setiap objek toko menyertakan atribut location dalam format GeoJSON Point.

```
{
 "name": "Toko Daur Ulang Berkah",
 "location": {
  "type": "Point",
  "coordinates": [112.63, -7.97]
 }
}
```

3. **Rendering Peta**: Frontend melakukan iterasi pada data yang diterima, lalu untuk setiap toko, sebuah *marker* (penanda) dibuat di peta pada koordinat yang sesuai. Ketika marker diklik, sebuah *popup* akan menampilkan informasi detail toko tersebut.

5.2.2. Menampilkan Lokasi Pengguna (Buyer/Distributor)

1. **Deteksi Lokasi**: Frontend menggunakan **HTML5 Geolocation API** (navigator.geolocation) untuk meminta izin dan mendapatkan koordinat latitude dan longitude dari perangkat pengguna saat ini.

2. **Penggunaan**:
    - **Untuk Pencarian Terdekat**: Koordinat ini dikirim sebagai parameter query ke endpoint GET /api/wastes/search?sort_by=nearby&latitude=...&longitude=... untuk mendapatkan hasil yang diurutkan berdasarkan jarak.
    - **Untuk Tampilan Peta**: Posisi pengguna saat ini dapat ditampilkan sebagai marker khusus (misalnya, lingkaran biru berdenyut) di peta untuk memberikan titik referensi visual.

5.2.3. Implementasi Live Tracking Posisi Distributor

Fitur ini memungkinkan buyer dan seller untuk memantau pergerakan distributor secara real-time saat pengiriman berlangsung.

5.2.3.1. Pengiriman Update Lokasi dari Aplikasi Distributor

1. Aplikasi di sisi distributor (bisa berupa web app atau mobile app) secara periodik (misalnya, setiap 30 detik) mendapatkan lokasi GPS perangkat.
2. Setiap kali lokasi baru didapat, aplikasi mengirimkan request **PATCH** ke endpoint http://127.0.0.1:8000/api/logistics/{logistics_id}/location dengan menyertakan token autentikasi dan body JSON berisi current_latitude dan current_longitude.

5.2.3.2. Penerimaan dan Pembaruan Lokasi di Peta Buyer/Seller (Polling)

1. Ketika buyer atau seller membuka halaman pelacakan transaksi, frontend akan memulai proses *polling*.
2. Secara periodik (misalnya, setiap 15-20 detik), frontend akan mengirimkan request **GET** ke endpoint http://127.0.0.1:8000/api/logistics/{logistics_id}/status.
3. Backend merespons dengan data logistik terkini, termasuk current_location distributor dalam format GeoJSON, serta distance_km dan estimated_delivery_time yang sudah dihitung ulang.
4. Frontend menerima data ini dan memperbarui posisi marker distributor di peta Leaflet serta menampilkan teks estimasi yang baru.
5. Proses polling ini terus berjalan hingga status transaksi berubah menjadi delivered atau cancelled.

5.3. Penggunaan Fungsi PostGIS

Backend memanfaatkan fungsi-fungsi dari ekstensi PostGIS untuk melakukan query geospasial yang efisien, dipermudah dengan package clickbar/laravel-magellan.

- **Penyimpanan Data**: Kolom seperti location (di tabel users, stores, recycling_facilities), delivery_location (di transactions), dan current_location (di logistics) menggunakan tipe data geography(Point, 4326).

- **Perhitungan Jarak (ST_Distance)**:
    - **Pada Fitur Pencarian Terdekat:** Saat pengguna meminta hasil berdasarkan sort_by=nearby, query Eloquent di WasteSearchController secara dinamis menambahkan perhitungan jarak menggunakan fungsi ST_Distance.

```
-- Contoh SQL yang dihasilkan oleh Laravel
SELECT ..., ST_Distance(stores.location, 'SRID=4326;POINT(lon lat)'::geography) / 1000 AS
distance_km
FROM wastes
JOIN stores ON ...
ORDER BY distance_km ASC;
```

- **Pada Fitur Live Tracking:** Di dalam LogisticsController, ST_Distance digunakan untuk menghitung jarak tersisa dari lokasi distributor saat ini ke lokasi tujuan pengiriman.

Indeks Spasial (GiST): Semua kolom bertipe geography diindeks menggunakan GiST (Generalized Search Tree). Ini secara drastis mempercepat query yang melibatkan filter atau pengurutan berbasis lokasi (ST_Distance, ST_DWithin, dll.), memastikan performa aplikasi tetap optimal meskipun data lokasi semakin banyak.

BAB 6 KEAMANAN

Keamanan data dan aplikasi adalah prioritas utama dalam pengembangan EcoBarter. Kami menerapkan pendekatan berlapis untuk memastikan integritas, kerahasiaan, dan ketersediaan data pengguna serta melindungi sistem dari ancaman umum.

6.1. Keamanan Autentikasi dan Otorisasi (Laravel Sanctum, Peran)

- **Autentikasi API:** Untuk mengamankan komunikasi antara frontend SPA dan backend, kami menggunakan **Laravel Sanctum**. Sanctum menyediakan sistem autentikasi yang kuat dan fleksibel, baik untuk SPA yang berjalan di domain yang sama (menggunakan cookie sesi yang aman) maupun untuk aplikasi pihak ketiga/mobile (menggunakan token API). Semua endpoint yang berisi data sensitif atau memerlukan aksi dari pengguna terdaftar diproteksi dengan middleware auth:sanctum.
- **Manajemen Peran:** Hak akses pengguna di dalam aplikasi diatur secara ketat menggunakan package **spatie/laravel-permission**. Setiap pengguna memiliki peran yang terdefinisi dengan jelas (Buyer, Seller, Distributor, Admin). Logika di backend (baik di controller maupun policy) akan selalu memeriksa peran pengguna sebelum mengizinkan mereka melakukan aksi tertentu, seperti membuat toko, mengkonfirmasi transaksi, atau menghapus data. Ini memastikan bahwa seorang Buyer tidak bisa melakukan aksi yang hanya diperuntukkan bagi Seller.

6.2. Proteksi terhadap Serangan Umum Web

Framework Laravel secara bawaan menyediakan beberapa mekanisme pertahanan yang kuat terhadap serangan umum:

- **Proteksi SQL Injection:** Dengan menggunakan **Eloquent ORM** dan **Query Builder** Laravel, semua input database secara otomatis di-*escape* menggunakan *parameter binding*. Ini secara efektif mencegah serangan SQL Injection karena input dari pengguna tidak pernah digabungkan langsung ke dalam string query mentah.
- **Proteksi Cross-Site Scripting (XSS):** Saat merender data di frontend menggunakan Blade (untuk bagian web tradisional) atau di framework SPA modern (seperti Vue/React), data yang ditampilkan secara default di-*escape*. Ini mencegah skrip berbahaya yang dimasukkan oleh pengguna untuk dieksekusi di browser pengguna lain.
- **Proteksi Cross-Site Request Forgery (CSRF):** Untuk semua rute web yang menggunakan formulir HTML, Laravel secara otomatis menghasilkan dan memvalidasi token CSRF. Ini memastikan bahwa setiap request POST, PUT, PATCH, atau DELETE berasal dari aplikasi kita sendiri, bukan dari situs eksternal yang berbahaya.

6.3. Keamanan API Endpoint

- **Validasi Input:** Setiap data yang masuk melalui API divalidasi secara ketat di sisi server menggunakan fitur Validasi Laravel. Aturan seperti required, string, numeric, max, exists:nama_tabel, dan in:nilai_valid diterapkan untuk memastikan data yang masuk sesuai

dengan format dan batasan yang diharapkan sebelum diproses lebih lanjut.
- **Rate Limiting:** Rute API secara default dilindungi oleh middleware throttle:api dari Laravel. Ini membatasi jumlah request yang dapat dibuat oleh satu IP dalam periode waktu tertentu, membantu melindungi server dari serangan *brute-force* atau request berlebihan yang dapat menyebabkan *denial of service* (DoS).

6.4. Keamanan Transmisi Data (HTTPS)

Aplikasi yang di-deploy secara online diwajibkan menggunakan protokol **HTTPS**. Dengan menginstal sertifikat SSL/TLS di server hosting, semua data yang dikirim antara browser klien dan server backend akan dienkripsi. Ini melindungi data sensitif seperti password, token autentikasi, dan informasi pribadi dari penyadapan (*man-in-the-middle attack*) saat transit melalui jaringan internet.

6.5. Privasi Data Pengguna

- **Password Hashing:** Password pengguna tidak pernah disimpan sebagai teks biasa. Laravel secara otomatis menggunakan algoritma hashing yang kuat (Bcrypt secara default) untuk mengenkripsi password saat disimpan di database.
- **Data Personal:** Akses terhadap data personal pengguna dibatasi secara ketat oleh sistem otorisasi berbasis peran. Data seperti alamat lengkap dan nomor telepon hanya ditampilkan kepada pihak yang berkepentingan dalam konteks transaksi (misalnya, distributor yang akan melakukan pengiriman).
- **Token API:** Token API Sanctum yang dihasilkan untuk pengguna juga di-*hash* di database, menambahkan lapisan keamanan ekstra.

6.7. Sumber Data Sekunder yang Digunakan

Sesuai dengan ketentuan lomba, aplikasi EcoBarter memanfaatkan data sekunder yang bersifat terbuka (*open source*) untuk memperkaya fungsionalitasnya:

1. **Peta Dasar:** Menggunakan data dari **OpenStreetMap (OSM)** yang diakses melalui *tile server* publik. OSM menyediakan data peta global yang gratis dan dapat digunakan secara bebas.
2. **API Rute dan Navigasi:** Menggunakan layanan demo dari **Open Source Routing Machine (OSRM)** (http://router.project-osrm.org). API ini digunakan untuk menghitung jarak rute jalan dan estimasi waktu tempuh untuk fitur live tracking distributor.
3. **Data Geospasial (Opsional Tambahan):** Untuk pengembangan lebih lanjut, data batas administrasi atau data demografis dapat diambil dari sumber data terbuka Indonesia seperti **Badan Pusat Statistik (BPS)** (bps.go.id) dan **Portal Data Indonesia** (data.go.id).

6.8. Relevansi Data dengan Tema dan Indonesia

- **Relevansi Tema:** Penggunaan data geospasial dari OpenStreetMap dan OSRM sangat relevan dengan tema *"Sustainable Digital Solutions"*, karena memungkinkan solusi logistik yang lebih

efisien (mencari rute terbaik) dan mendukung ekonomi sirkular dengan menghubungkan penjual dan pembeli limbah berdasarkan lokasi geografis.

- **Konteks Indonesia:** Data peta yang digunakan mencakup seluruh wilayah Indonesia secara detail. Fitur pencarian "terdekat" secara langsung menggunakan lokasi pengguna di Indonesia untuk memberikan hasil yang relevan secara lokal, mendukung UMKM daur ulang dan pengepul di seluruh negeri.

6.9. Cara Data Diproses dan Disimpan

- **Penyimpanan Data Geospasial:** Semua data yang memiliki komponen geografis (lokasi pengguna, toko, distributor, dan tujuan pengiriman) disimpan menggunakan tipe data geography(Point, 4326) di database PostgreSQL yang didukung oleh ekstensi **PostGIS**.
- **Pemrosesan Data Live Tracking:**
    1. Aplikasi di perangkat distributor mengirimkan koordinat latitude dan longitude saat ini ke backend API.
    2. Backend menerima koordinat ini, menyimpannya di kolom current_location pada tabel logistics.
    3. Backend kemudian memanggil API OSRM dengan mengirimkan koordinat saat ini dan koordinat tujuan (yang tersimpan di tabel transactions).
    4. Respons dari OSRM (berupa jarak dan durasi) diproses dan disimpan ke dalam kolom distance_km dan duration_minutes di tabel logistics.
- **Serialisasi Data:** Saat data (terutama yang mengandung lokasi) dikirim dari backend ke frontend melalui API, package clickbar/laravel-magellan secara otomatis mengubah tipe data PostGIS menjadi format **GeoJSON** standar. Format ini sangat mudah diproses oleh pustaka peta JavaScript seperti Leaflet.js untuk ditampilkan sebagai marker di peta.

BAB 7: KONFIGURASI DAN PENGELOLAAN HOSTING SERTA DOMAIN

9.1. Layanan Hosting yang Digunakan

- **Penyedia:** Microsoft Azure
- **Layanan:** Azure App Service (Linux)
- **Paket:** Basic (B1)
- **Spesifikasi:**
    - 1 instance
    - 2 vCores (D2s_v3)
    - 8 GiB RAM
    - 128 GiB Storage

9.2. Alasan Pemilihan Layanan Hosting

- **Performa:** Azure menyediakan infrastruktur cloud yang stabil dengan opsi auto-scaling dan resources yang cukup untuk kebutuhan pengembangan hingga produksi.
- **Ketersediaan:** Lokasi deployment berada di Southeast Asia, mendekati target pengguna untuk latency rendah.
- **Keamanan:** Mendukung deployment HTTPS, pengaturan firewall, dan manajemen identitas melalui Azure AD.

9.3. Konfigurasi Server / Environment

- **Runtime:** PHP 8.4 (Linux)
- **Database:** Azure PostgreSQL Flexible Server
    - **Versi PostgreSQL:** 12.20

    - **Konfigurasi:** 2 vCores, 8 GiB RAM
    - **Ekstensi:** PostGIS belum allow-listed secara default oleh Azure. Error extension "postgis" is not allow-listed for "azure_pg_admin" muncul saat dicoba.
- **Ekstensi PHP:** mbstring, bcmath, pdo, pdo_pgsql
- **Build Tools:** Vite + Node.js (diinstall manual untuk kompilasi assets CSS/JS)

9.4. Nama Domain dan Pengelolaannya

- **Domain Default:** https://laravel-webgis-btgkadgpe9gwdndf.southeastasia-01.azurewebsites.net
- **Custom Domain:** Belum ditambahkan (opsional melalui menu Azure > Custom Domain)

9.5. Implementasi HTTPS (SSL/TLS Certificate)

- **Status:** Aktif secara default dari Azure App Service
- **Masalah yang Ditemui:** Mixed Content Warning ketika asset CSS/JS dimuat melalui protokol http.
- **Solusi:** Pastikan base URL dan asset di-serve menggunakan protokol https. Tambahkan di .env:

APP_URL=https://laravel-webgis-btgkadgpe9gwdndf.southeastasia-01.azurewebsites.net

Dan pastikan konfigurasi Vite memuat asset dengan protokol aman.

9.6. Proses Deployment

- **Metode:** GitHub Actions (CI/CD)
- **Workflow:** deploy.yml
    - Checkout repo
    - Setup PHP dan dependencies
    - Composer install (tanpa dev)

    - Build Vite (manual menggunakan Node.js, jika diaktifkan)
    - Zip dan deploy ke App Service
- **Deployment Provider:** GitHub Actions
- **Terakhir Sukses:** 6 Juni 2025, 10:36:47 AM