

SPESIFIKASI KEBUTUHAN PERANGKAT LUNAK (SKPL)

Nama Aplikasi: EcoLedger (Immutable Carbon Tracker)

Tema: SDG 13 - Climate Action

Jenis Basis Data: NoSQL (Document-based)

Tanggal: 25 Desember 2024

Status: Draft

1. PENDAHULUAN

1.1 Latar Belakang

Perubahan iklim telah menjadi tantangan global yang mendesak, sejalan dengan target Sustainable Development Goals (SDG) 13 mengenai Climate Action. Transparansi dalam pelaporan emisi karbon adalah kunci untuk mencapai target iklim yang ambisius.

Namun, dunia saat ini dihadapkan pada masalah serius: **Greenwashing** – praktik manipulasi data untuk membuat organisasi atau individu terlihat lebih ramah lingkungan daripada kenyataannya.

EcoLedger hadir sebagai solusi inovatif yang menyediakan platform pencatatan jejak karbon individual dan organisasi dengan fitur keamanan integritas data berbasis konsep **Cryptographic Hashing Chain** (mirip Blockchain). Dengan pendekatan basis data NoSQL yang fleksibel, sistem ini memastikan setiap data emisi tidak dapat dimanipulasi tanpa meninggalkan jejak yang terdeteksi.

1.2 Tujuan Sistem

Sistem EcoLedger dirancang dengan tujuan sebagai berikut:

1. **Menyediakan Platform Pencatatan Emisi Harian** – Memungkinkan pengguna mencatat aktivitas sehari-hari dari kategori Transportasi, Konsumsi Listrik, dan Pola Konsumsi Makanan.
2. **Implementasi Database NoSQL** – Menggunakan MongoDB untuk fleksibilitas dalam penyimpanan dan pengolahan data yang beragam.
3. **Menjamin Integritas Data** – Mengimplementasikan mekanisme Cryptographic Hashing Chain untuk mencegah manipulasi data (CRUD operations) tanpa jejak audit.
4. **Menyajikan Visualisasi Data Statistik** – Memberikan insights visual mengenai tren emisi karbon melalui dashboard interaktif.
5. **Mendukung Verifikasi Data** – Memungkinkan system/admin untuk melakukan checking hash dan memvaliditas keseluruhan rantai data.

1.3 Ruang Lingkup

Dokumen ini merespesifikasi:

- Kebutuhan fungsional (Functional Requirements)
- Kebutuhan non-fungsional (Non-Functional Requirements)
- Desain basis data NoSQL
- Desain antarmuka pengguna (UI/UX)
- Mekanisme keamanan integritas data
- Implementasi API dan logika bisnis

Tidak termasuk dalam ruang lingkup:

- Implementasi sistem pembayaran (billing)
- Integrasi media sosial
- Mobile application native (hanya web-based)

2. DESKRIPSI UMUM SISTEM

2.1 Perspektif Produk

EcoLedger adalah aplikasi web berbasis cloud yang memungkinkan pengguna untuk mencatat, melacak, dan memverifikasi jejak karbon pribadi atau organisasi mereka. Sistem menggunakan arsitektur client-server dengan database NoSQL sebagai backbone.

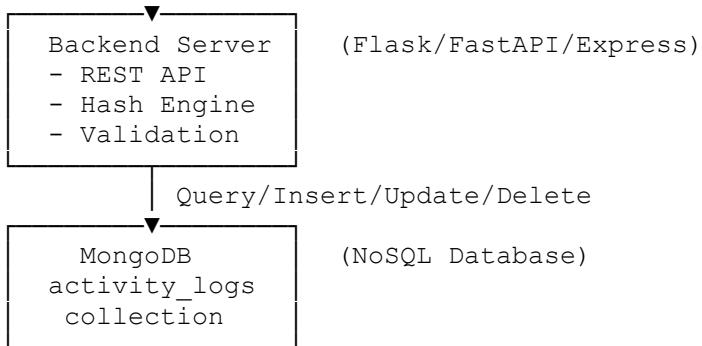
Arsitektur Umum:

text

Web Browser

(HTML, CSS, JavaScript)

HTTP/HTTPS



2.2 Karakteristik Pengguna

Tipe Pengguna	Deskripsi	Kemampuan
User Umum	Individu atau karyawan organisasi yang ingin mencatat emisi karbon harian mereka	<ul style="list-style-type: none"> - Menambah aktivitas baru - Melihat riwayat emisi - Memperbarui data aktivitas - Melihat dashboard statistik - Export laporan
System/Admin	Administrator sistem yang memiliki akses penuh	<ul style="list-style-type: none"> - Melakukan verifikasi hash chain - Monitoring integritas data - Pengelolaan user - Audit trail

2.3 Lingkungan Operasi (Tech Stack)

Frontend

- **HTML5** – Markup structure
- **CSS3 / Bootstrap 5** – Styling dan responsive design
- **JavaScript (ES6+)** – Interaksi dan validasi client-side
- **Chart.js** – Visualisasi data (Pie Chart, Line Chart)

Backend

- **Framework:** Flask (Python) **atau** Express.js (Node.js)
- **Database:** MongoDB (Document-based NoSQL), Apache Cassandra untuk h
- **Authentication:** JWT (JSON Web Tokens)
- **Hashing:** SHA-256 (built-in di Python/Node.js)

External Services (Optional)

- **Carbon Interface API** – Konversi aktivitas ke emisi karbon (jika diperlukan)
- **Hosting:** Heroku, Vercel, atau AWS

Browser Support

- Chrome (Latest)

- Firefox (Latest)
- Safari (Latest)
- Edge (Latest)

3. KEBUTUHAN FUNGSIONAL (FUNCTIONAL REQUIREMENTS)

3.1 Daftar Kebutuhan Fungsional

ID	Nama Fitur	Deskripsi	Tipe CRUD	Prioritas
FR-01	Input Aktivitas	Pengguna memasukkan data aktivitas (misal: Jarak tempuh mobil, kWh listrik, gram makanan). Sistem menghitung estimasi CO2 otomatis berdasarkan faktor emisi standar.	CREATE	High
FR-02	Hashing Engine	Sistem men-generate Hash unik (SHA-256) berdasarkan data input + Hash data sebelumnya. Setiap data baru memiliki current_hash dan mereferensikan previous_hash.	System Logic	Critical
FR-03	Lihat Riwayat Emisi	Menampilkan daftar lengkap aktivitas dalam bentuk tabel kronologis beserta status validitas data (Valid ✓ / Invalid ✗).	READ	High
FR-04	Koreksi Data	Pengguna mengedit data input yang salah. Sistem akan memberikan peringatan bahwa rantai hash akan rusak ("Tampered") dan menampilkan dampak pada data selanjutnya.	UPDATE	Medium
FR-05	Hapus Data	Menghapus log aktivitas tertentu. Mengakibatkan data selanjutnya dalam rantai menjadi invalid (demonstrasi keamanan integritas data).	DELETE	Medium
FR-06	Dashboard Statistik	Menampilkan ringkasan emisi total, grafik Pie Chart (Kategori Emisi), Line Chart (Tren Emisi Harian/Mingguan/Bulanan).	READ	High
FR-07	Verifikasi Hash Chain	Fitur admin untuk melakukan checking ulang semua hash di database. Sistem menghitung ulang hash dari data yang ada dan membandingkan dengan hash tersimpan.	System Logic	Critical

ID	Nama Fitur	Deskripsi	Tipe CRUD	Prioritas
FR-08	Export Laporan	Memungkinkan user mengunduh laporan emisi dalam format PDF atau CSV.	READ	Low
FR-09	Filter & Pencarian	Pencarian aktivitas berdasarkan tanggal, kategori, atau kata kunci.	READ	Medium
FR-10	Notifikasi Integritas**	Alert visual ketika terdeteksi manipulasi data (Hash mismatch) dengan penjelasan teknis untuk admin.	System Logic	High
FR-11	System Audit Log	Sistem secara otomatis mencatat setiap interaksi user (Login, Create, Update, Delete) ke dalam log yang detail dan berkecepatan tinggi.	CASSANDRA	High
FR-12	View Audit Trail	Admin dapat melihat riwayat aktivitas user secara real-time untuk pemantauan keamanan.	Cassandra	High

3.2 Detail Kebutuhan Fungsional

FR-01: Input Aktivitas

User Story: Sebagai pengguna, saya ingin mencatat aktivitas harian saya agar sistem dapat menghitung emisi karbon saya.

Alur Kerja:

1. Pengguna mengakses halaman "Tambah Aktivitas"
2. Pilih kategori aktivitas:
 - **Transportasi** (Kendaraan, Jarak, Jumlah Penumpang)
 - **Listrik** (kWh yang digunakan)
 - **Makanan** (Jenis makanan, porsi/gram)
3. Isi detail dan deskripsi
4. Sistem otomatis menghitung emisi berdasarkan faktor emisi
5. Sistem menyimpan data dengan hashing
6. Tampilkan konfirmasi sukses

Kriteria Penerimaan:

- ✓ Form validation di client-side
- ✓ Error handling jika data kosong
- ✓ CO2 calculation akurat
- ✓ Data disimpan dengan previous_hash dan current_hash

FR-02: Hashing Engine (Core Security)

Deskripsi Teknis:

Mekanisme ini adalah jantung integritas data. Setiap dokumen dalam koleksi activity_logs memiliki struktur hash chain.

Algoritma Hashing:

text

Input:

- timestamp (ISO 8601)

- activity_type (string)
- carbon_emission (float)
- previous hash (dari dokumen sebelumnya)

Process:

1. Ambil current_hash dari dokumen terakhir
 2. Simpan ke field previous_hash dokumen baru
 3. Buat string: SHA256(timestamp + activity_type + carbon_emission + previous_hash)
 4. Simpan hasil sebagai current_hash

Output:

- current hash (64 karakter hexadecimal)

Contoh:

text

Dokumen 1 (Genesis) :

```
timestamp: 2024-12-26T08:00:00Z  
activity_type: Transportasi  
carbon_emission: 4.75  
previous_hash: 0000000000000000  
current_hash: 5a1b2c3d4e5f6a7b8
```

Dokumen 2:

```
timestamp: 2024-12-26T12:00:00Z  
activity_type: Listrik  
carbon_emission: 8.5  
previous_hash: 5a1b2c3d4e5f6a7b  
current_hash: a1b2c3d4e5f6a7b8c
```

Jika dokumen 1 dimanipulasi, current_hash akan berubah, tetapi dokumen 2 masih mereferensikan hash lama → **Terdeteksi sebagai CORRUPTED**.

FR-03: Lihat Riwayat Emisi

User Story: Sebagai pengguna, saya ingin melihat semua aktivitas emisi saya dalam satu tempat dengan informasi lengkap.

Tampilan:

- Tabel dengan kolom: Tanggal, Kategori, Detail, Emisi (kg), Hash ID, Status
 - Sorting: Default descending (terbaru di atas)
 - Pagination: 10 data per halaman

Kolom Status:

- ✓ **Valid (Hijau)** - Hash match, data tidak dimanipulasi
 - ✗ **Invalid (Merah)** - Hash mismatch, data mungkin dimanipulasi
 - ⚡ **Orphaned (Kuning)** - Data sebelumnya dihapus, rantai putus

FB-04 : Koreksi Data

User Story: Sebagai pengguna, saya ingin memperbaiki data yang salah karena typo atau kesalahan input.

Alur Kerja:

1. User klik tombol "Edit" pada aktivitas
 2. Form pre-filled dengan data lama
 3. User ubah data
 4. Sistem menampilkan warning:

⚠ "Jika Anda mengubah data ini, hash akan berubah dan mungkin merusak integritas rantai berikutnya. Anda yakin?"

5. User konfirmasi
 6. Sistem melakukan:
 - Update dokumen
 - Recalculate current_hash
 - Mark dokumen selanjutnya sebagai INVALID

Kriteria Penerimaan:

- ✓ Warning ditampilkan sebelum update
 - ✓ Dokumen selanjutnya otomatis marked INVALID
 - ✓ Audit log mencatat perubahan (optional)
-

FR-05: Hapus Data

User Story: Sebagai pengguna, saya ingin menghapus aktivitas yang tidak relevan.

Behavior:

1. User klik "Hapus"
2. Konfirmasi dialog
3. Sistem melakukan:
 - Delete dokumen
 - Update previous_hash dokumen selanjutnya ke dokumen sebelumnya
 - Recalculate current_hash dokumen selanjutnya
 - Jika field index digunakan, re-index semua dokumen setelahnya

Dampak:

- Dokumen yang sebelumnya INVALID tetap INVALID
 - Dokumen yang sudah ada menjadi INVALID karena previous_hash berubah
-

FR-06: Dashboard Statistik

User Story: Sebagai pengguna, saya ingin melihat visualisasi emisi saya untuk memahami pola dan area yang bisa dikurangi.

Komponen Dashboard:**A. Summary Cards:**

- Total Emisi (Bulan Ini): XX kg CO2
- Emisi Hari Ini: X kg CO2
- Kategori Terbesar: [Category] - X% dari total
- Status Integritas: Valid/Invalid

B. Pie Chart - Distribusi Emisi per Kategori

- Transportasi: 45%
- Listrik: 35%
- Makanan: 20%

C. Line Chart - Tren Emisi Harian

- X-axis: Tanggal (7 hari terakhir)
- Y-axis: Emisi (kg CO2)
- Tooltip: Detail emisi per hari

D. Statistics Table:

- Rata-rata emisi per hari
 - Emisi maksimal
 - Emisi minimal
-

FR-07: Verifikasi Hash Chain

User Story: Sebagai admin, saya ingin memverifikasi integritas seluruh rantai hash untuk mendeteksi manipulasi data.

Proses Verifikasi:

text

```
FOR EACH dokumen IN activity_logs (sorted by index):
    1. Hitung ulang hash: calculated_hash = SHA256(timestamp + activity_type +
carbon_emission + previous_hash)
    2. Bandingkan dengan stored_hash = dokumen.current_hash
    3. IF calculated_hash ≠ stored_hash:
        MARK dokumen sebagai CORRUPTED
        MARK semua dokumen berikutnya sebagai ORPHANED
        LOG event ke audit trail
    4. ELSE:
        MARK dokumen sebagai VALID
```

Output Verifikasi:

- Total dokumen: X
- Valid: Y
- Corrupted: Z
- Orphaned: W
- Timestamp verifikasi terakhir

FR-08: Export Laporan

User Story: Sebagai pengguna, saya ingin mengunduh laporan emisi saya dalam format standar.

Format Tersedia:

1. **PDF** - Laporan formal dengan grafik
2. **CSV** - Data raw untuk analisis lanjutan

Konten:

- Periode laporan (dari-sampai)
- Total emisi
- Breakdown per kategori
- Tabel detail (untuk CSV)
- Generated on timestamp

FR-09: Filter & Pencarian

User Story: Sebagai pengguna, saya ingin mencari aktivitas spesifik dengan cepat.

Fitur Filter:

- **Berdasarkan Tanggal:** Range picker (dari-sampai)
- **Berdasarkan Kategori:** Dropdown (Transportasi, Listrik, Makanan)
- **Berdasarkan Kata Kunci:** Text input (cari di field user_note)
- **Berdasarkan Status:** Valid/Invalid/All

FR-10: Notifikasi Integritas

User Story: Sebagai pengguna/admin, saya ingin mendapat alert jika terjadi anomali integritas data.

Jenis Notifikasi:

1. **Saat Verifikasi Gagal:**

A "Terdeteksi manipulasi data pada aktivitas [Tanggal]. Hash tidak cocok. Hubungi admin untuk investigasi."

2. **Saat Melakukan Update/Delete:**

i "Rantai hash telah diperbarui. Data yang mengikuti mungkin berubah statusnya."

3. **Saat Admin Login:**

Alert jika ada dokumen dengan status CORRUPTED

4. KEBUTUHAN NON-FUNGSIONAL (NON-FUNCTIONAL REQUIREMENTS)

ID	Kategori	Persyaratan	Target
NFR-01	Performa	Response time untuk GET request	< 500ms
NFR-02	Performa	Response time untuk POST/PUT request	< 1000ms
NFR-03	Skalabilitas	Support minimal 1000 user	Baseline

ID	Kategori	Persyaratan	Target
		concurrent	
NFR-04	Keamanan	HTTPS encryption untuk semua komunikasi	Mandatory
NFR-05	Keamanan	Hashing algorithm (SHA-256) minimal	Mandatory
NFR-06	Keamanan	User authentication dengan JWT	Mandatory
NFR-07	Availability	Uptime sistem	99.5% (per bulan)
NFR-08	Usability	Mobile-responsive design	Bootstrap 5 breakpoints
NFR-09	Reliability	Backup database otomatis	Daily
NFR-10	Compliance	Compliance dengan GDPR (data privacy)	Pseudonymization

5. DESAIN BASIS DATA (NoSQL SCHEMA)

5.1 Struktur Database MongoDB

Database Name: eco_ledger_db

Collections:

1. activity_logs - Log aktivitas dengan hashing
2. users - Data pengguna
3. audit_trail (optional) - Catatan perubahan

5.2 Apache Cassandra (Column-based) Digunakan untuk menyimpan log aktivitas dalam volume besar (*Write-Heavy*).

- Keyspace: eco_logs
- Table: activity_audit

Column Name	Data Type	Description
user_id	TEXT	Partition Key (Untuk pengelompokan data per user)
activity_time	TIMESTAMP	Clustering Key (DESC) - Untuk urutan waktu
action_type	TEXT	Jenis aksi (misal: "CREATE_LOG", "LOGIN_SUCCESS")
description	TEXT	Detail aktivitas (misal: "Input 50km motor")
ip_address	TEXT	Alamat IP pengguna

5.3 Koleksi: activity_logs

Struktur Dokumen:

Field Explanation:

Field	Type	Deskripsi
_id	ObjectId	Primary key MongoDB (auto-generated)
user_id	ObjectId	Reference ke user document
timestamp	String (ISO 8601)	Waktu aktivitas terjadi
activity_type	String	Kategori (Transportasi, Listrik, Makanan)
details	Object	Data spesifik sesuai tipe aktivitas
carbon_emission	Float	Hasil kalkulasi emisi (kg CO2e)
unit	String	Satuan emisi

Field	Type	Deskripsi
user_note	String	Catatan pengguna (optional)
index	Integer	Urutan dokumen dalam rantai
previous_hash	String	Hash dokumen sebelumnya (64 hex chars)
current_hash	String	Hash dokumen ini (64 hex chars)
status	String	VALID / INVALID / ORPHANED
verified_at	String (ISO 8601)	Waktu terakhir verifikasi

5.4 Koleksi: users

```
json
{
  "_id": ObjectId("64f7a1b2c3d4e5f6g7h8i9j0"),
  "username": "john_eco",
  "email": "john@example.com",
  "password_hash": "hashed_password_here",
  "profile": {
    "full_name": "John Doe",
    "organization": "PT Green Initiative"
  },
  "role": "user", // user, admin
  "created_at": "2024-12-01T08:00:00Z",
  "updated_at": "2024-12-26T10:00:00Z",
  "is_active": true
}
```

5.5 Koleksi: audit_trail (Optional)

```
json
{
  "_id": ObjectId("64f8a1b2c3d4e5f6g7h8i9j0"),
  "user_id": ObjectId("64f7a1b2c3d4e5f6g7h8i9j0"),
  "action": "CREATE", // CREATE, UPDATE, DELETE, VERIFY
  "entity": "activity_logs",
  "entity_id": ObjectId("64f8a1b2c3d4e5f6g7h8i9j0"),
  "changes": {
    "old_value": null,
    "new_value": { "carbon_emission": 4.75 }
  },
  "timestamp": "2024-12-26T10:30:00Z",
  "ip_address": "192.168.1.1"
}
```

5.6 Index Recommendations

```
javascript
// Untuk query performa
db.activity_logs.createIndex({ "user_id": 1, "timestamp": -1 })
```

```

db.activity_logs.createIndex({ "activity_type": 1 })
db.activity_logs.createIndex({ "status": 1 })
db.activity_logs.createIndex({ "current_hash": 1 })

db.users.createIndex({ "username": 1 }, { unique: true })
db.users.createIndex({ "email": 1 }, { unique: true })

```

6. IMPLEMENTASI API & LOGIKA BISNIS

6.1 Sumber Data Emisi (Carbon Calculation)

Faktor Emisi Standar (dapat disesuaikan berdasarkan region):

Transportasi

Jenis Kendaraan	Faktor Emisi	Unit
Mobil Bensin	0.190	kg CO ₂ / km
Mobil Diesel	0.170	kg CO ₂ / km
Motor	0.065	kg CO ₂ / km
Bus	0.089	kg CO ₂ / km / penumpang
Pesawat	0.255	kg CO ₂ / km / penumpang
Kereta	0.041	kg CO ₂ / km / penumpang

Listrik

Sumber	Faktor Emisi	Unit
Grid Nasional (Indonesia)	0.85	kg CO ₂ / kWh
Terbarukan	0.10	kg CO ₂ / kWh

Makanan

Kategori	Faktor Emisi	Unit
Beef	27.0	kg CO ₂ / kg
Chicken	6.9	kg CO ₂ / kg
Vegetables	2.0	kg CO ₂ / kg
Dairy	3.2	kg CO ₂ / kg

Rumus Kalkulasi:

text

carbon_emission = user_input × faktor_emisi × multiplier

Contoh:

```
text
Input: 25 km (Mobil Bensin)
Calculation: 25 km × 0.190 kg CO2/km = 4.75 kg CO2
```

6.2 REST API Endpoints**Authentication**

```
text
```

```
POST /api/auth/register
  Body: { username, email, password, full_name }
  Response: { token, user_id }
```

```
POST /api/auth/login
  Body: { email, password }
  Response: { token, user_id, role }
```

```
POST /api/auth/logout
  Headers: Authorization: Bearer <token>
  Response: { message: "Logout successful" }
```

Activity Management

```
text
```

```
GET /api/activities
  Headers: Authorization: Bearer <token>
  Query: ?user_id=X&category=Transportasi&from=2024-12-01&to=2024-12-31
  Response: [{ activity data with hashing }]
```

```
POST /api/activities
  Headers: Authorization: Bearer <token>
  Body: {
    activity_type: "Transportasi",
    details: { vehicle: "Mobil Bensin", distance_km: 25, passengers: 1 },
    user_note: "Perjalanan ke Kampus"
  }
  Response: { _id, carbon_emission, current_hash, status: "VALID" }
```

```
GET /api/activities/:id
  Headers: Authorization: Bearer <token>
  Response: { activity document }
```

```
PUT /api/activities/:id
  Headers: Authorization: Bearer <token>
  Body: { Updated fields }
  Response: { Updated document, warning: "Hash chain may be affected" }
```

```
DELETE /api/activities/:id
  Headers: Authorization: Bearer <token>
  Response: { message: "Activity deleted, subsequent records marked INVALID" }
```

Dashboard & Reports

```
text
```

```
GET /api/dashboard/summary
  Headers: Authorization: Bearer <token>
  Response: {
    total_emission: 150.5,
    today_emission: 8.5,
    largest_category: "Transportasi",
    integrity_status: "VALID"
  }
```

```
GET /api/dashboard/statistics
```

```
Headers: Authorization: Bearer <token>
Query: ?period=weekly | ?period=monthly
Response: {
    pie_chart_data: [ { category, percentage } ],
    line_chart_data: [ { date, emission } ]
}

GET /api/reports/export
Headers: Authorization: Bearer <token>
Query: ?format=pdf|csv&from=2024-12-01&to=2024-12-31
Response: File download

Admin Only - Verification
text
POST /api/admin/verify-hash-chain
Headers: Authorization: Bearer <token> (admin role)
Response: {
    total_records: 100,
    valid: 95,
    corrupted: 5,
    orphaned: 0,
    verification_timestamp: "2024-12-26T11:00:00Z",
    corrupted_records: [ { _id, current_hash, calculated_hash } ]
}

GET /api/admin/audit-trail
Headers: Authorization: Bearer <token> (admin role)
Query: ?user_id=X&action=CREATE|UPDATE|DELETE&from=2024-12-01&to=2024-12-31
Response: [ audit entries ]
```

6.3 Mekanisme Integritas Data (Cryptographic Hashing Chain)

Saat CREATE (Tambah Aktivitas)

```

# Step 5: Insert document
new_doc = {
    "user_id": user_id,
    "timestamp": timestamp,
    "activity_type": activity_type,
    "details": details,
    "carbon_emission": carbon_emission,
    "user_note": user_note,
    "index": index,
    "previous_hash": previous_hash,
    "current_hash": current_hash,
    "status": "VALID",
    "verified_at": timestamp
}

result = db.activity_logs.insert_one(new_doc)
return result.inserted_id

Saat READ/VERIFY (Cek Integritas)
python
# Pseudocode (Python)

def verify_hash_chain(user_id=None):
    # Get all docs (filter by user_id if provided)
    query = {"user_id": user_id} if user_id else {}
    docs = db.activity_logs.find(query).sort("index", 1)

    results = {
        "total": 0,
        "valid": 0,
        "corrupted": 0,
        "orphaned": 0,
        "corrupted_records": []
    }

    for doc in docs:
        results["total"] += 1

        # Recalculate hash
        hash_input =
f"{doc['timestamp']}{doc['activity_type']}{doc['carbon_emission']}{doc['previous_ha
sh']}"
        calculated_hash = sha256(hash_input).hexdigest()

        # Compare
        if calculated_hash == doc["current_hash"]:
            results["valid"] += 1
            # Update status to VALID if it was checked
            db.activity_logs.update_one(
                {"_id": doc["_id"]},
                {"$set": {"status": "VALID", "verified_at": get_iso_timestamp()}}
            )
        else:
            results["corrupted"] += 1
            results["corrupted_records"].append({
                "_id": str(doc["_id"]),
                "stored_hash": doc["current_hash"],
                "calculated_hash": calculated_hash
            })
            # Mark as corrupted

```

```

        db.activity_logs.update_one(
            {"_id": doc["_id"]},
            {"$set": {"status": "INVALID"}}
        )
        # Mark all subsequent docs as ORPHANED
        db.activity_logs.update_many(
            {"user_id": user_id, "index": {"$gt": doc["index"]}},
            {"$set": {"status": "ORPHANED"}}
        )
        results["orphaned"] += len(list(docs.clone().skip(docs.index)))
    break # Stop verification at first corrupted doc

return results
Saat UPDATE (Ubah Data)
python
def update_activity(activity_id, updated_fields):
    # Get the document to update
    doc = db.activity_logs.find_one({"_id": ObjectId(activity_id)})

    # Update carbon emission if details changed
    if "details" in updated_fields:
        updated_fields["carbon_emission"] = calculate_emission(
            doc["activity_type"],
            updated_fields["details"]
        )

    # Recalculate current hash
    hash_input =
f"{doc['timestamp']}{doc['activity_type']}{updated_fields.get('carbon_emission',
doc['carbon_emission'])}{doc['previous_hash']}"
    new_hash = sha256(hash_input).hexdigest()

    # Update document
    updated_fields["current_hash"] = new_hash
    updated_fields["status"] = "VALID"
    updated_fields["updated_at"] = get_iso_timestamp()

    db.activity_logs.update_one(
        {"_id": ObjectId(activity_id)},
        {"$set": updated_fields}
    )

    # Mark all subsequent documents as INVALID (because previous_hash changed)
    next_docs = db.activity_logs.find({
        "user_id": doc["user_id"],
        "index": {"$gt": doc["index"]}
    })

    for next_doc in next_docs:
        db.activity_logs.update_one(
            {"_id": next_doc["_id"]},
            {"$set": {"status": "INVALID"}}
        )

    return {
        "message": "Activity updated",
        "warning": "Subsequent documents marked as INVALID"
    }
Saat DELETE (Hapus Data)

```

7. DESAIN ANTARMUKA (UI/UX WIREFRAME)

7.1 Navigasi Utama

text.

[Main Content Area]

7.2 Halaman Dashboard (Home)

text

Dashboard > Home

 Total Emisi
150.5 kg CO2

 Hari Ini
8.5 kg CO2

Kategori Terbesar: Transportasi (45%)

Status Integritas: ✓ VALID

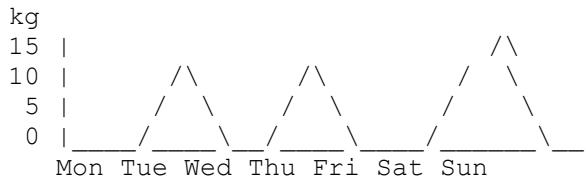
Verifikasi Terakhir: 2024-12-26 10:35

[+ Tambah Aktivitas]

Pie Chart - Distribusi

Transportasi	Listrik
45%	35%
Makanan	20%

Line Chart - Trend Emisi (7 hari)



7.3 Halaman Riwayat (Ledger)

text

Dashboard > Riwayat Emisi

Filter: [Kategori ▼] [Dari ▼] [Sampai ▼] [Cari...]

Status: [Semua ▼]

Tanggal

Kategori

Detail

Emisi
(kg)

26 Des 2024 10:30	Transportasi (Kampus)	Mobil 25 km (Kampus)	4.75 ✓ Valid
26 Des 2024 08:00	Listrik (Rumah)	10 kWh (Rumah)	8.50 ✓ Valid
25 Des 2024 19:30	Makanan	Ayam 500g	3.45 ✗ Invalid (edited)
25 Des 2024 07:00	Transportasi	Motor 15 km	0.98 ⚠ Orphaned (chain broke)

[Edit] [Delete] [< Prev] [Next >]

7.4 Halaman Tambah Aktivitas

text

Dashboard > Tambah Aktivitas

Pilih Kategori:

- Transportasi
- Listrik
- Makanan

Form: TRANSPORTASI

Jenis Kendaraan: [Mobil Bensin ▼]
Jarak Tempuh (km): [____ 25 ____]
Jumlah Penumpang: [____ 1 ____]
Catatan (opsional): [Perjalanan ke Kampus....]
Estimasi Emisi: 4.75 kg CO2

[Cancel] [Simpan Aktivitas]

7.5 Dialog Edit Aktivitas (Dengan Warning)

text

Edit Aktivitas

⚠ PERINGATAN:
Mengubah data ini akan merusak integritas rantai hash. Semua aktivitas setelahnya mungkin menjadi INVALID. Anda yakin ingin melanjutkan?

[X] Jangan tampilkan lagi

[Batal] [Lanjutkan Edit]

7.6 Halaman Admin - Verifikasi Hash Chain

text

Admin > Verifikasi Integritas Data												
Verifikasi Terakhir: 2024-12-26 10:35:00 [Jalankan Verifikasi Sekarang]												
Hasil:												
Total Dokumen: 250 ✓ Valid: 245 ✗ Corrupted: 3 ⚠ Orphaned: 2												
Dokumen Rusak:												
<table border="1"><thead><tr><th>ID</th><th>Stored Hash</th><th>Calc Hash</th></tr></thead><tbody><tr><td>64f8a1b2c3</td><td>5a1b2c3d4e5f..</td><td>a1b2c3d4e5f..</td></tr><tr><td>64f8a1b2c4</td><td>6a2b3c4d5e6f..</td><td>b2c3d4e5f6g..</td></tr><tr><td>64f8a1b2c5</td><td>7a3b4c5d6e7f..</td><td>c3d4e5f6g7h..</td></tr></tbody></table>	ID	Stored Hash	Calc Hash	64f8a1b2c3	5a1b2c3d4e5f..	a1b2c3d4e5f..	64f8a1b2c4	6a2b3c4d5e6f..	b2c3d4e5f6g..	64f8a1b2c5	7a3b4c5d6e7f..	c3d4e5f6g7h..
ID	Stored Hash	Calc Hash										
64f8a1b2c3	5a1b2c3d4e5f..	a1b2c3d4e5f..										
64f8a1b2c4	6a2b3c4d5e6f..	b2c3d4e5f6g..										
64f8a1b2c5	7a3b4c5d6e7f..	c3d4e5f6g7h..										
[Export Laporan] [Hubungi Pengguna] [Investigate]												

8. TIMELINE & DELIVERABLES

8.1 Milestone & Jadwal

Fase	Deskripsi	Duration	Target Selesai
Fase 1	Design & Setup Database	2 minggu	01 Jan 2025
Fase 2	Backend API Development (CRUD + Hashing)	2 minggu	08 Jan 2025
Fase 3	Frontend Development (UI/UX)	2 minggu	15 Jan 2025
Fase 4	Integration & Testing	1 minggu	22 Jan 2025
Fase 5	Deployment & Documentation	1 minggu	29 Jan 2025

8.2 Deliverables

- ✓ SRS (Spesifikasi Kebutuhan Perangkat Lunak)
- ✓ Database Design Document
- ✓ API Documentation (Swagger/OpenAPI)
- ✓ Functional Source Code (Frontend + Backend)

- ✓ Unit Test & Integration Test Results
- ✓ User Manual & Admin Guide
- ✓ Deployment Guide
- ✓ Demo Video

9. RISIKO & MITIGATION

Risiko	Probabilitas	Impact	Mitigation
Kompleksitas Hashing Chain	Medium	High	Mulai dengan implementasi simple, tambah complexity bertahap
Database Performance	Medium	Medium	Implementasi indexing sejak awal, load testing
User Adoption	Low	Medium	Provide good documentation & tutorials
Security Vulnerability	Low	High	Security review oleh expert, OWASP top 10 compliance
Integration Issues	Medium	Medium	Regular integration testing, early API testing

10. KESIMPULAN

EcoLedger adalah sistem pencatatan emisi karbon dengan fokus pada **integritas data** dan **transparansi** menggunakan prinsip Cryptographic Hashing Chain berbasis NoSQL. Sistem ini dirancang untuk mendukung SDG 13 (Climate Action) dengan memberikan kepercayaan kepada pengguna bahwa data mereka tidak dapat dimanipulasi tanpa jejak yang terdeteksi.

Implementasi dilakukan dengan tech stack modern (MongoDB, Flask/Express, React/Vanilla JS) dan mengikuti best practices dalam software engineering.

REFERENSI DOKUMEN

- ISO/IEC 25010:2015 – Software Product Quality Standards
- OWASP Top 10 2023 – Web Application Security
- UN SDG 13 – Climate Action
- MongoDB Best Practices Documentation
- SHA-256 Cryptographic Algorithm Specification
- Carbon Accounting Standards (GHG Protocol)

Dokumen ini merupakan DRAFT dan terbuka untuk revisi dan feedback.

Prepared by: [Nama Tim]

Date: 25 Desember 2024

Version: 1.0