**Langkah – Langkah Perbaikan**

**‘Pengelolaan Transaksi yang Belum Tersimpan (Session)’**

“**Masalah**:

Saat melakukan transaksi, data transaksi disimpan sementara di session. Jika session berakhir atau pengguna tidak menyimpan transaksi dengan benar, data transaksi bisa hilang. Hal ini menyebabkan ketidakakuratan dalam manajemen transaksi dan stok barang.

**Solusi**:

Implementasikan fitur autosave atau draft yang secara otomatis menyimpan transaksi ke backend (database) secara berkala atau setiap kali ada perubahan. Dengan demikian, meskipun session berakhir atau pengguna tidak menyimpan transaksi secara manual, data transaksi tetap tersimpan dan bisa dilanjutkan di kemudian hari. “

1. **Membuat Model dan Tabel untuk Transaksi Draft**

Pertama, kita perlu membuat model dan migrasi untuk menyimpan transaksi yang belum tersimpan atau dalam status draft.

1. Buat model TransaksiDraft dan migrasi terkait.

*php artisan make:model TransaksiDraft -m*

1. Edit file migrasi yang baru dibuat (database/migrations/xxxx\_xx\_xx\_create\_transaksi\_drafts\_table.php):

*use Illuminate\Database\Migrations\Migration;*

*use Illuminate\Database\Schema\Blueprint;*

*use Illuminate\Support\Facades\Schema;*

*class CreateTransaksiDraftsTable extends Migration*

*{*

*public function up()*

*{*

*Schema::create('transaksi\_drafts', function (Blueprint $table) {*

*$table->id();*

*$table->unsignedBigInteger('user\_id'); // ID pengguna yang membuat transaksi*

*$table->json('barang'); // Data barang yang dibeli dalam format JSON*

*$table->decimal('total\_harga', 15, 2);*

*$table->decimal('diskon', 5, 2)->default(0);*

*$table->decimal('total\_setelah\_diskon', 15, 2);*

*$table->timestamps();*

*$table->foreign('user\_id')->references('id')->on('users')->onDelete('cascade');*

*});*

*}*

*public function down()*

*{*

*Schema::dropIfExists('transaksi\_drafts');*

*}*

*}*

1. Jalankan migrasi:

*php artisan migrate*

1. **Membuat Controller untuk Menangani Transaksi Draft**
2. Buat controller TransaksiDraftController:

*php artisan make:controller TransaksiDraftController*

1. Edit app/Http/Controllers/TransaksiDraftController.php:

*<?php*

*namespace App\Http\Controllers;*

*use Illuminate\Http\Request;*

*use App\Models\TransaksiDraft;*

*use Illuminate\Support\Facades\Auth;*

*class TransaksiDraftController extends Controller*

*{*

*// Menyimpan atau memperbarui draft transaksi*

*public function saveDraft(Request $request)*

*{*

*$user = Auth::user();*

*$barang = $request->input('barang'); // Array barang*

*$totalHarga = $request->input('total\_harga');*

*$diskon = $request->input('diskon', 0);*

*$totalSetelahDiskon = $request->input('total\_setelah\_diskon');*

*// Cari apakah sudah ada draft untuk user ini*

*$draft = TransaksiDraft::where('user\_id', $user->id)->first();*

*if ($draft) {*

*// Update draft yang sudah ada*

*$draft->barang = json\_encode($barang);*

*$draft->total\_harga = $totalHarga;*

*$draft->diskon = $diskon;*

*$draft->total\_setelah\_diskon = $totalSetelahDiskon;*

*$draft->save();*

*} else {*

*// Buat draft baru*

*TransaksiDraft::create([*

*'user\_id' => $user->id,*

*'barang' => json\_encode($barang),*

*'total\_harga' => $totalHarga,*

*'diskon' => $diskon,*

*'total\_setelah\_diskon' => $totalSetelahDiskon,*

*]);*

*}*

*return response()->json(['message' => 'Draft transaksi disimpan']);*

*}*

*// Mengambil draft transaksi untuk user*

*public function getDraft()*

*{*

*$user = Auth::user();*

*$draft = TransaksiDraft::where('user\_id', $user->id)->first();*

*if ($draft) {*

*return response()->json([*

*'barang' => json\_decode($draft->barang),*

*'total\_harga' => $draft->total\_harga,*

*'diskon' => $draft->diskon,*

*'total\_setelah\_diskon' => $draft->total\_setelah\_diskon,*

*]);*

*} else {*

*return response()->json(['message' => 'Tidak ada draft transaksi'], 404);*

*}*

*}*

*// Menghapus draft transaksi setelah selesai*

*public function deleteDraft()*

*{*

*$user = Auth::user();*

*$draft = TransaksiDraft::where('user\_id', $user->id)->first();*

*if ($draft) {*

*$draft->delete();*

*return response()->json(['message' => 'Draft transaksi dihapus']);*

*} else {*

*return response()->json(['message' => 'Tidak ada draft transaksi'], 404);*

*}*

*}*

*}*

1. **Menambahkan Routes untuk Transaksi Draft**

Tambahkan route untuk mengakses fungsi-fungsi di TransaksiDraftController

1. Edit file routes/web.php atau routes/api.php (sesuai dengan struktur aplikasi Anda):

*use App\Http\Controllers\TransaksiDraftController;*

*// Route untuk API (jika menggunakan API)*

*Route::middleware('auth')->group(function () {*

*Route::post('/transaksi-draft/save', [TransaksiDraftController::class, 'saveDraft']);*

*Route::get('/transaksi-draft/get', [TransaksiDraftController::class, 'getDraft']);*

*Route::delete('/transaksi-draft/delete', [TransaksiDraftController::class, 'deleteDraft']);*

*});*

1. **Implementasi Autosave di Frontend**

Gunakan JavaScript untuk secara otomatis menyimpan draft transaksi ke backend setiap kali ada perubahan pada transaksi.

1. Tambahkan JavaScript berikut di halaman transaksi Anda:

*let transaksiTimer;*

*// Fungsi untuk menyimpan draft transaksi*

*function saveDraftTransaksi() {*

*const barang = getBarangTransaksi(); // Fungsi untuk mendapatkan data barang yang dibeli*

*const totalHarga = calculateTotalHarga(); // Fungsi untuk menghitung total harga*

*const diskon = getDiskon(); // Fungsi untuk mendapatkan diskon*

*const totalSetelahDiskon = calculateTotalSetelahDiskon(totalHarga, diskon);*

*fetch('/transaksi-draft/save', {*

*method: 'POST',*

*headers: {*

*'Content-Type': 'application/json',*

*'X-CSRF-TOKEN': document.querySelector('meta[name="csrf-token"]').getAttribute('content') // Pastikan CSRF token ada*

*},*

*body: JSON.stringify({*

*barang: barang,*

*total\_harga: totalHarga,*

*diskon: diskon,*

*total\_setelah\_diskon: totalSetelahDiskon*

*})*

*})*

*.then(response => response.json())*

*.then(data => {*

*console.log(data.message);*

*})*

*.catch(error => {*

*console.error('Error:', error);*

*});*

*}*

*// Fungsi untuk memulai timer autosave*

*function startAutosave() {*

*transaksiTimer = setInterval(saveDraftTransaksi, 30000); // Autosave setiap 30 detik*

*}*

*// Panggil fungsi startAutosave saat halaman transaksi dimuat*

*document.addEventListener('DOMContentLoaded', (event) => {*

*startAutosave();*

*});*

*// Hentikan autosave saat halaman transaksi ditutup atau dibersihkan*

*window.addEventListener('beforeunload', (event) => {*

*clearInterval(transaksiTimer);*

*});*

Catatan:

* Fungsi getBarangTransaksi(), calculateTotalHarga(), getDiskon(), dan calculateTotalSetelahDiskon() harus diimplementasikan sesuai dengan logika aplikasi Anda untuk mendapatkan data transaksi saat ini.
* Pastikan CSRF token ditambahkan jika menggunakan Laravel untuk keamanan permintaan POST.

1. **Mengambil dan Melanjutkan Draft Transaksi saat Halaman Dimuat**

Ketika pengguna membuka halaman transaksi, periksa apakah ada draft transaksi yang tersimpan dan muat data tersebut.

1. Tambahkan JavaScript berikut untuk mengambil draft transaksi saat halaman dimuat:

*document.addEventListener('DOMContentLoaded', (event) => {*

*// Mulai autosave*

*startAutosave();*

*// Ambil draft transaksi jika ada*

*fetch('/transaksi-draft/get', {*

*method: 'GET',*

*headers: {*

*'Content-Type': 'application/json',*

*'X-CSRF-TOKEN': document.querySelector('meta[name="csrf-token"]').getAttribute('content')*

*}*

*})*

*.then(response => {*

*if (response.ok) {*

*return response.json();*

*} else {*

*throw new Error('Tidak ada draft transaksi');*

*}*

*})*

*.then(data => {*

*loadDraftTransaksi(data);*

*})*

*.catch(error => {*

*console.log(error.message);*

*});*

*});*

*// Fungsi untuk memuat draft transaksi ke tampilan*

*function loadDraftTransaksi(data) {*

*const barang = data.barang;*

*barang.forEach(item => {*

*addBarangKeTampilan(item); // Fungsi untuk menambahkan barang ke tampilan transaksi*

*});*

*setTotalHarga(data.total\_harga);*

*setDiskon(data.diskon);*

*setTotalSetelahDiskon(data.total\_setelah\_diskon);*

*}*

Catatan:

* Fungsi addBarangKeTampilan(), setTotalHarga(), setDiskon(), dan setTotalSetelahDiskon() harus diimplementasikan sesuai dengan logika aplikasi Anda untuk memuat data transaksi ke tampilan.

1. **Menghapus Draft Transaksi Setelah Transaksi Selesai**

Setelah transaksi selesai disimpan secara permanen, hapus draft transaksi dari database.

1. Setelah transaksi disimpan ke database utama, panggil endpoint untuk menghapus draft transaksi.

*function selesaiTransaksi(transaksi) {*

*fetch('/selesaikan-transaksi', {*

*method: 'POST',*

*headers: {*

*'Content-Type': 'application/json',*

*'X-CSRF-TOKEN': document.querySelector('meta[name="csrf-token"]').getAttribute('content')*

*},*

*body: JSON.stringify(transaksi)*

*})*

*.then(response => response.json())*

*.then(data => {*

*alert('Transaksi berhasil diselesaikan');*

*// Hapus draft transaksi*

*fetch('/transaksi-draft/delete', {*

*method: 'DELETE',*

*headers: {*

*'Content-Type': 'application/json',*

*'X-CSRF-TOKEN': document.querySelector('meta[name="csrf-token"]').getAttribute('content')*

*}*

*})*

*.then(response => response.json())*

*.then(data => {*

*console.log(data.message);*

*// Bersihkan tampilan transaksi*

*clearTransaksiTampilan();*

*})*

*.catch(error => {*

*console.error('Error:', error);*

*});*

*})*

*.catch(error => {*

*console.error('Error:', error);*

*});*

*}*

Catatan:

* Fungsi clearTransaksiTampilan() harus diimplementasikan untuk membersihkan tampilan transaksi setelah berhasil diselesaikan.

1. **Menampilkan Daftar Draft Transaksi untuk Pengguna (Opsional)**

Jika pengguna memiliki lebih dari satu draft transaksi atau ingin melihat riwayat draft, tambahkan fitur ini sesuai kebutuhan.

1. Tambahkan tampilan di frontend untuk menampilkan daftar draft transaksi yang ada.
2. Implementasikan fungsi di backend untuk mengambil semua draft transaksi jika diperlukan.

* **Testing**
* Simpan Draft Secara Manual:
* Tambahkan barang ke transaksi dan perhatikan apakah draft transaksi disimpan secara otomatis setiap 30 detik.
* Cek di database apakah data draft tersimpan dengan benar.
* Lanjutkan Draft Transaksi:
* Tutup halaman transaksi tanpa menyelesaikan transaksi.
* Buka kembali halaman transaksi dan periksa apakah draft transaksi dimuat dengan benar.
* Selesaikan Transaksi:
* Selesaikan transaksi yang telah dimuat dari draft.
* Pastikan draft transaksi dihapus dari database setelah transaksi selesai.
* Cek Validasi:
* Pastikan bahwa jika ada perubahan pada transaksi sebelum draft disimpan, data yang disimpan adalah data terbaru.
* Uji dengan Beberapa Pengguna:
* Lakukan pengujian dengan beberapa pengguna untuk memastikan bahwa draft transaksi disimpan dan diambil sesuai dengan pengguna yang bersangkutan.
* **Hasil Perbaikan**
* Draft Transaksi Tersimpan: Transaksi yang belum selesai akan tersimpan secara otomatis ke database sebagai draft.
* Lanjutkan Transaksi: Pengguna dapat melanjutkan transaksi yang belum selesai tanpa kehilangan data, meskipun session berakhir atau pengguna menutup halaman.
* Pengalaman Pengguna Lebih Baik: Pengguna merasa lebih aman dan nyaman karena data transaksi tidak hilang secara tidak sengaja.
* Integrasi yang Lebih Baik: Sistem transaksi menjadi lebih terintegrasi dengan backend, memastikan konsistensi data.