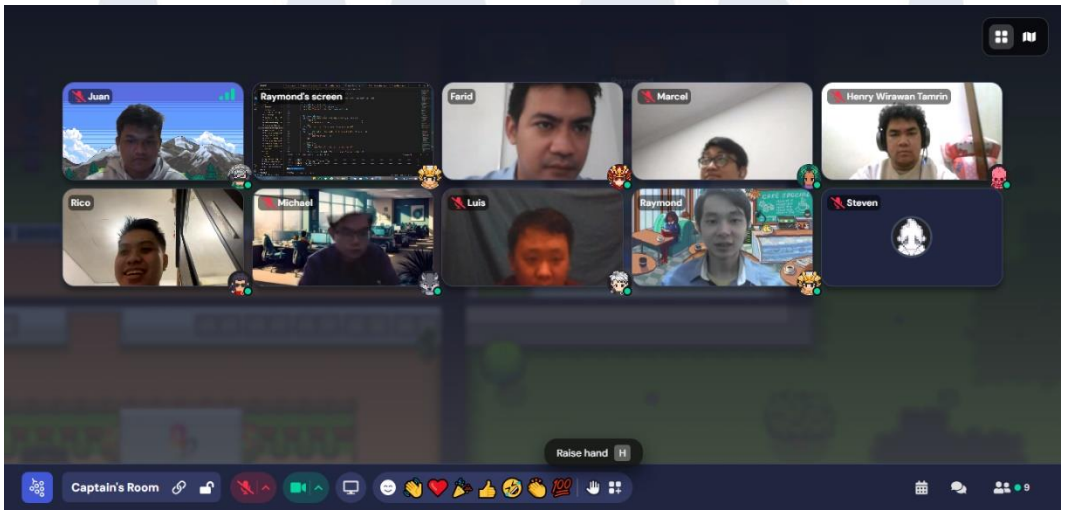
BAB III

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Dalam periode magang MBKM track 2 ini, penulis diposisikan pada *software developer* (*back-end developer*) di PT Ritzproject Sinergi Visitama. Pada posisi ini penulis memiliki peran untuk mengembangan kembali sistem *e-commerce* yang sebelumnya telah dibuat oleh periode magang sebelumnya. Sistem *e-commerce* ini merupakan salah satu product dari perusahaan Ritzproject Sinergi Visitama. Pada kesempatan kali ini penulis melanjutkan sistem *e-commerce* dikarenakan pada periode magang sebelumnya telah melakukan pembuatan dengan produk ini sehingga penulis diberikan tugas untuk melanjutkan dan mengembangan kembali sistem *e-commerce* secara universal sebagai produk yang bisa dipakai oleh perusahaan untuk mendapatkan klien sehingga jika suatu saat nanti terdapat klien yang tertarik dengan produk ini setelah magang berakhir maka keinginan klien setelah periode magang akan dikerjakan kembali oleh peserta magang berikutnya ataupun karyawan tetap dari perusahaan Ritzproject Sinergi Visitama.



Gambar 3. 1 Tim Magang di PT Ritzproject Visitama Sinergi

Gambar 3. 2 Tim Magang di PT Ritzproject Visitama Sinergi (I)

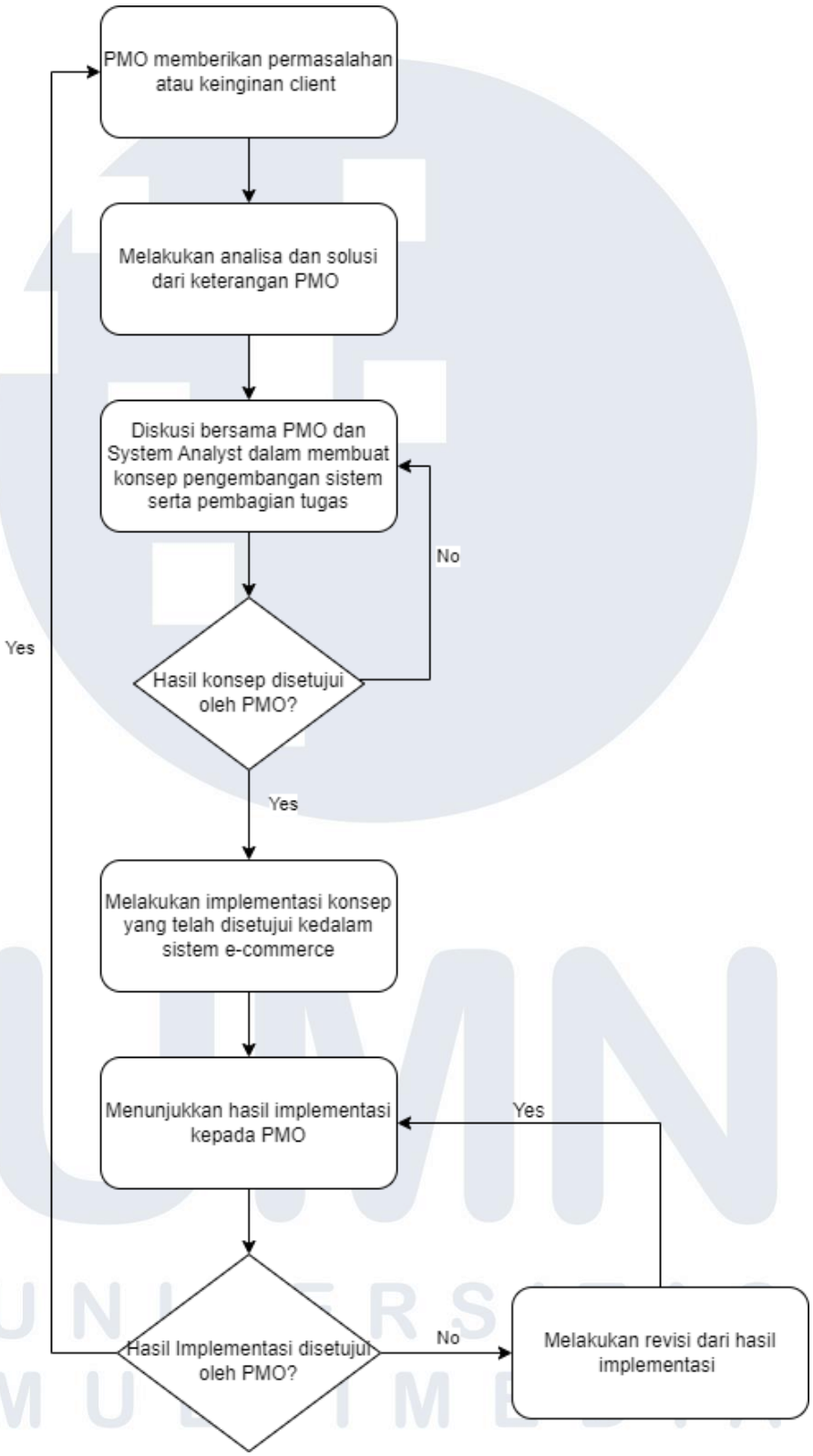
Dalam menjalankan kerja magang di PT Ritzproject Sinergi Visitama, penulis secara langsung dibimbing dan diawasi oleh *Project Management Officer* (PMO) PT Ritzproject Sinergi Visitama yaitu Bapak Farid Ananda. Dengan mendapatkan bimbingan secara langsung maka penulis mendapatkan arahan dalam pengembangan sistem *e-commerce* ketika *system analyst* menjelaskan mengenai menu tambahan yang telah dibuatkan kepada penulis. Periode magang ini penulis bergabung dengan tim yang memiliki divisi yang berbeda yaitu *system analyst*, *software developer* (*front-end developer*), *software developer* (*back-end developer*), dan *qualiity assurance*. Dalam satu tim terdiri dari 8 orang yaitu 1 orang *system analyst*, 1 orang *front-end developer*, 5 orang *back-end developer*, 1 orang *quality assurance* dapat terlihat pada gambar 3.1 dan 3.2. Selaku 1 tim ini kami bekerja sama untuk mengembangkan kembali sistem *e-commerce* ini yang sudah sempat dibuat perserta magang pada periode *bacth* sebelumnya.

Dalam ketiga peran ini yang dibagi menjadi 8 orang maka kami bekerja sama dalam hal sebagai berikut:

1. *System analyst* dan *software developer* mendesain mockup web, mendetailkan dokumen *spesification program*.
2. *System analyst* dan *software developer* (*back-end developer*) mendesain *database* yang akan digunakan *back-end developer* dalam mengembangkan sistem *back-end* menggunakan *framework* Laravel.

3. *Software developer* (*back-end developer*) dalam mengembangkan sistem e-commerce bagian *back-end* penulis dan tim *back-end developer* menggunakan bahasa pemograman PHP dengan *framework* Laravel.
4. *Quality assurance* melakukan pengecekan automasi dari hasil yang telah dikerjakan oleh *software developer* (*back-end developer*) agar sesuai dengan dokumen *spesification program*.

Penulis selaku dengan 4 orang *back-end developer* lainnya memilki beberapa perbedaan dalam melakukan pekerjaannya. Pada awalnya *back-end developer* ini melakukan pekerjaan secara bersama pada bagian pengembangan login admin dan master toko tapi setelah itu terdapat pembagian tugas setiap *back-end developer*. Penulis mendapatkan tugas pada bagian transaksi website *e-commerce* yang mengurus diskon bagi member, ada event, ulang tahun, dan lainnya. Adapun alur kerja yang diterapkan di PT Ritzproject Sinergi Visitama selaku penulis sebagai intern pada posisi *software developer* (*back-end developer*) pada gambar 3.3



Gambar 3. 3 Alur kerja di PT Ritzproject Sinergi Visitama

Alur kerja pada gambar 3.3 merupakan alur kerja yang dilakukan oleh penulis selaku peserta magang MBKM track 2 dalam selama melakukan kerja magang di Perusahaan Ritzproject Sinergi Visitama. Pertama *project management officer* (PMO) akan memberikan sebuah kebutuhan dari *system analyst* selaku *project management officer* kepada kami sehingga kami bisa melakukan analisis dan solusi dari kebutuhan klien. Setelah melakukan analisis maka akan disampaikan terlebih dahulu kepada PMO. Hal ini diperlukan karena analisis dan solusi perlu disetujui oleh PMO terlebih dahulu. Jika tidak setujui maka akan melakukan diskusi secara bersama-sama dengan PMO hingga disetujui oleh PMO selaku *supervisor* juga. Terkadang PMO memberikan masukan dan arahan yang membantu pengembagan sistem *e-commerce*. Dalam melakukan diskusi bersama ini juga membahas peran masing-masing kami untuk mengembangkan sistem *e-commerce*.

Dari hasil diskusi yang telah disetujui maka akan melakukan implementasi sistem *e-commerce* sesuai dengan peran kami masing-masing dan melakukan report hasil kerja kami kepada *project management officer* (PMO) pada meet berikutnya untuk melihat perkembangannya. Jika implementasi telah tidak disetujui oleh PMO maka akan melakukan revisi dan menunjukkan hasil revisi kepada PMO. Jika disetujui maka PMO akan memberikan task baru untuk dikerjakan kembali kepada kami.

3.2 Tugas dan Uraian Kerja Magang

Peran dari penulis selaku posisi *back-end developer* pada PT. Ritzproject Sinergi Visatama yaitu melakukan pengembangan pada sistem *e-commerce* yang dapat menjadi produk dari perusahaan PT. Ritzproject Sinergi Visatama.

Tabel 3. 1 Tabel Kegiatan Magang Perusahaan Ritzproject Sinergi Visitama

No	Pekerjaan	Minggu	Tanggal Mulai	Tanggal Selesai
Breafing Project & Team				
1.	Perkenalan team yang bergabung dengan PT.	1	26 Juni 2023	30 Juni 2023

	Ritzproject Sinergi Visatama, project yang ingin dikerjakan dan melakukan installasi *tools* serta mempelajari *tools* yang digunakan.			
2.	Mereview dan mempelajari *document* serta adanya penambahan tabel database untuk project yang dikerjakan	2	3 Juli 2023	7 Juli 2023
3.	Mempersiapkan dan membuat tampilan *mockup*	2-3	8 Juli 2023	13 Juli 2023
Development Admin				
4.	Membuat *back-end* pada bagian admin master Toko	3-4	14 Juli 2023	21 Juli 2023
5.	Membuat *back-end* untuk Login Admin	5	24 Juli 2023	29 Juli 2023
Development Transaction				
6.	Memperbaiki *bug* pada bagian *wishlist* dan *cart*	6	31 Juli 2023	3 Agustus 2023
7.	Membuat *back-end* pada bagian *cart* hanya diperbolehkan 1 toko saat *user* melakukan transaksi	6-7	4 Agustus 2023	9 Agustus 2023
8.	Membuat *back-end* berupa transaction for discount event	7-8	10 Agustus 2023	19 Agustus 2023
9.	Membuat *back-end* berupa *transaction for discount birthday*	9	21 Agustus 2023	26 Agustus 2023
10.	Membuat *back-end* berupa *transaction for discount member*	10	28 Agustus 2023	2 September 2023

11.	Membuat *back-end* berupa *transaction point for member*	11	4 September 2023	9 September 2023
12.	Membuat *back-end* untuk *transaction pay term*	12	11 September 2023	16 September 2023
13.	Membuat *back end* untuk menjumlahkan point dan menampilkan point tersebut di *home*	13	18 September 2023	23 September 2023
14.	Memperbaiki point member dan menambahkan point per barang	14	25 September 20239	26 September 2023
15.	Memperbaiki tampilan *home* pada bagian *flashsale* menjadi *slick slide* yang telah dibuat oleh *front-end*	14	27 September 2023	29 September 2023
16.	Memperbaiki tampilan dan coding *cart*	14	30 September 2023	30 September 2023
17.	Membantu dalam *return product* pada bagian admin	15	2 Oktober 2023	2 Oktober 2023
18.	Memperbaiki transaksi *discount* sesuai item di cart	15	2 Oktober 2023	3 Oktober 2023
19.	Menambahkan field di *database* dan memperbaiki agar bisa memasukkan hasil diskon di transaksi detail	15	4 Oktober 2023	5 Oktober 2023
20.	Menambahkan kondisi pada tampilan *transaction payment* dan Memeriksa serta memperbaiki bagian *xendit controller* yang mengalami bug	15	6 Oktober 2023	6 Oktober 2023

21.	Memeriksa kembali validasi dari pengerjaan yang telah dilakukan sebelumnya seperti cartblade, master header, master headerblade_edit, mastertoko_edit, update header,	15-16	7 Oktober 2023	10 Oktober 2023
Development Return (Admin)				
22.	Membantu dalam pembuatan pengembalian point, penambahan item dan penambahan saldo setelah user *return item*	16	11 Oktober 2023	12 Oktober 2023
23.	Menambahkan field di database berupa imember, ibirthday, ishipping_fee pada tabel ttransaction_hdr dan tpayment dan memperbaiki agar bisa memasukkan hasil diskon member, birthday, shipping fee di transaksi detail	16	13 Oktober 2023	13 Oktober 2023
24.	Memperbaiki tabel ttransaksi_hdr, ttransaction_dtl, tretur dan memperbaiki pengembalian point, penambahan item dan penambahan saldo setelah user return item	17	14 Oktober 2023	18 Oktober 2023
25.	Memperbaiki *pay-term* diakibatkan adanya perubahan harga setelah return product	17	19 Oktober 2023	21 Oktober 2023
Bug Fixing				

26.	Bug fixing cart bagian quantity, search bar, dan membantu dalam mengubah tampilan return transaction menggunakan function pagination dan menambahkan code untuk max return	18	23 Oktober2023	27 Oktober 2023
User Accept Test (UAT) Result				
27.	Phase Feedback UAT Result	19-22	30 Oktober 2023	25 November 2023

Penulis selaku *back-end developer* melakukan pengembangan pada bagian login admin, master toko, dan transaksi pada website dan melakukan pengembangan lainnya yang mendukung pengembangan website *e-commerce*.

3.2.1 Breafing project & Team

3.2.1.1 Introduction Team dan project

1) 26 Juni – 30 Juni

Pada kegiatan yang ini, *developer* melakukan pengenalan team yang tergabung pada magang batch 2. Selain itu, *developer* juga diperkenalkan mengenai perusahaan PT. Ritzproject Sinergi Visitama mulai dari visi-misi perusahaan. Kegiatan yang terjadi pada minggu ini juga melakukan pengenalan project yang ingin dikembangkan dan membahas pengembangan yang akan dilakukan selama batch periode 2 ini.

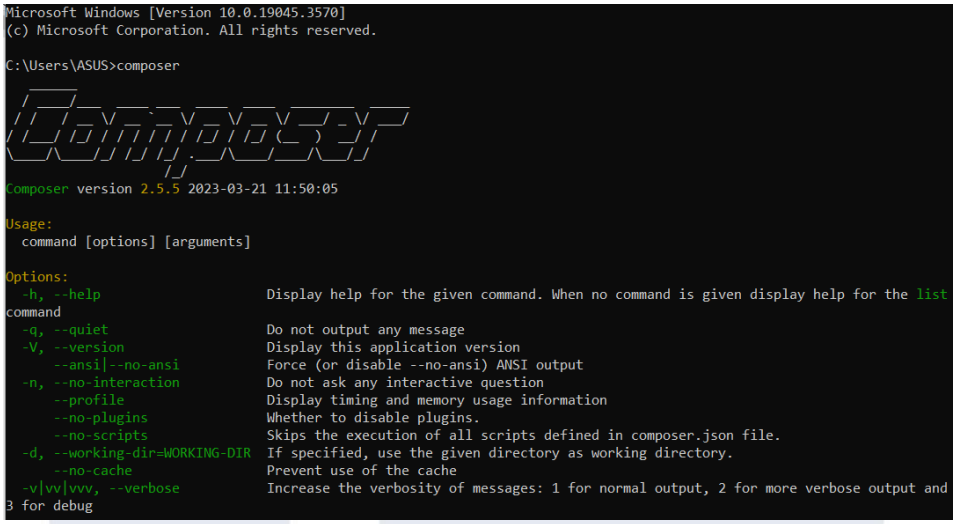
Gambar 3. 4 Visual Studio Code

Sumber: Techcelup.com

Selain melakukan pengenalan, *developer* juga melakukan instalasi beberapa *tools* yang digunakan selama mengerjakan project ini. *Tools* yang digunakan untuk mengerjakan project ini yaitu visual studio code yang digunakan untuk sebagai *tool editor code* yang mendukung berbagai bahasa pemograman serta *tools* ini bisa membangun dan mendebug aplikasi web secara mudah.

A Dependency Manager for PHP

Latest: **2.6.5** (changelog)



Gambar 3. 5 Composer Version

Developer melakukan penginstalan composer. Composer ini merupakan *tools* yang perlu didownload karena *tools* ini bisa mengelola dan mengatur *library*, *framework* yang menggunakan bahasa pemograman PHP. Gambar 3.5 merupakan hasil bahwa laptop saya telah berhasil melakukan instalasi composer dengan versi 2.5.5.

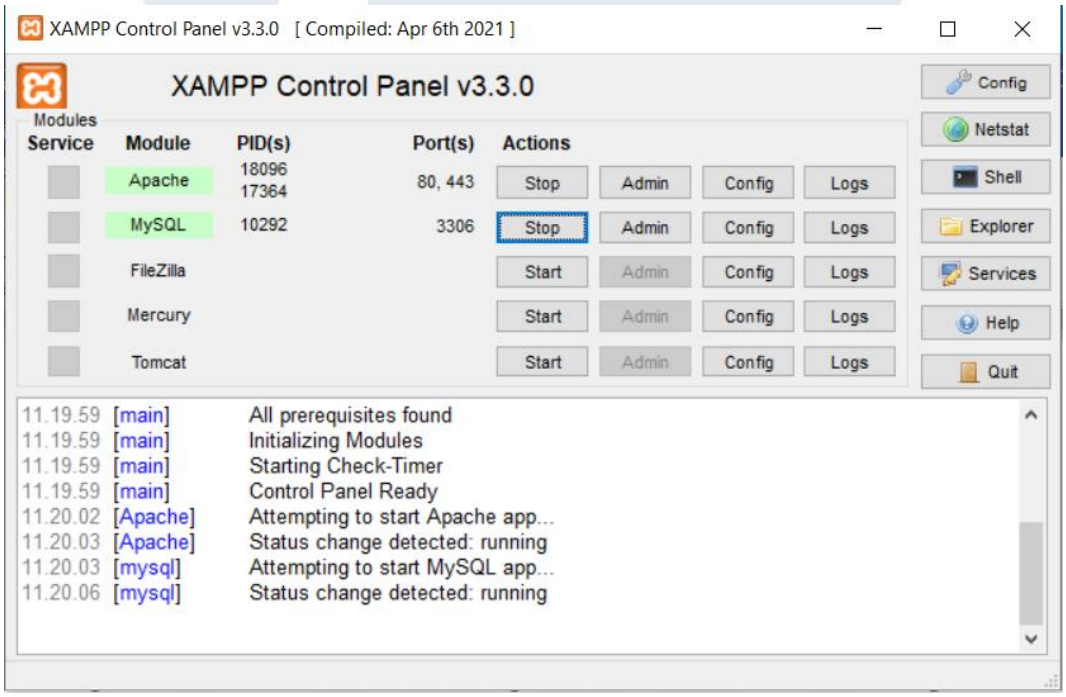
Gambar 3. 6 DBeaver Tool

Sumber: enterprisedb.com

Developer juga melakukan penginstalan Dbeaver yang merupakan *software* untuk mengelola dan mengakses berbagai jenis *database*. Dengan menggunakan *tools* ini akan lebih memudahkan *developer* dalam mengelola *database* dan tidak perlu melakukan pengelolaan *database* di phpmyadmin.

Gambar 3. 7 Laravel Version

Setelah itu, *developer* juga melakukan instalasi *framework* Laravel sebagai *framework* yang digunakan pada project ini. Laravel ini merupakan *framework* PHP yang menyediakan berbagai fitur dalam pembuatan *website* yang berbasis pada PHP. Gambar 3.7 adalah versi Laravel *framewok* yang berhasil *developer* instal yaitu versi 9.52.9.

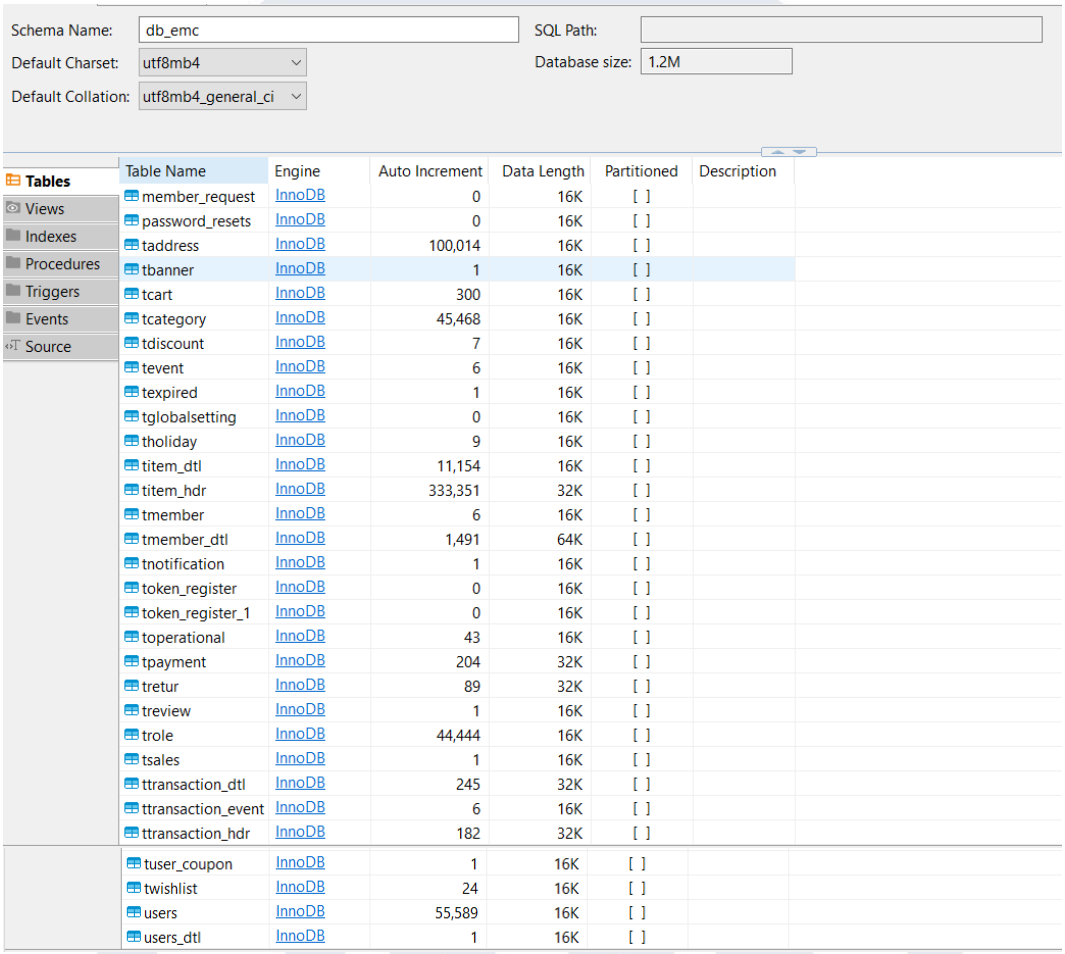


Gambar 3. 8 XAMPP Control Panel

Tidak hanya itu saja dalam menjalankan project ini juga memerlukan XAMPP sebagai *web server local host* yang dapat digunakan untuk mengembangkan dan merancang *website* secara local. Gambar 3.8 merupakan XAMPP yang *developer* gunakan versi 3.3.0. Dengan menjalankan XAMPP maka *website* baru bisa dijalankan.

3.2.1.2 Pengenalan Document Project

1) 3 Juli – 7 Juli



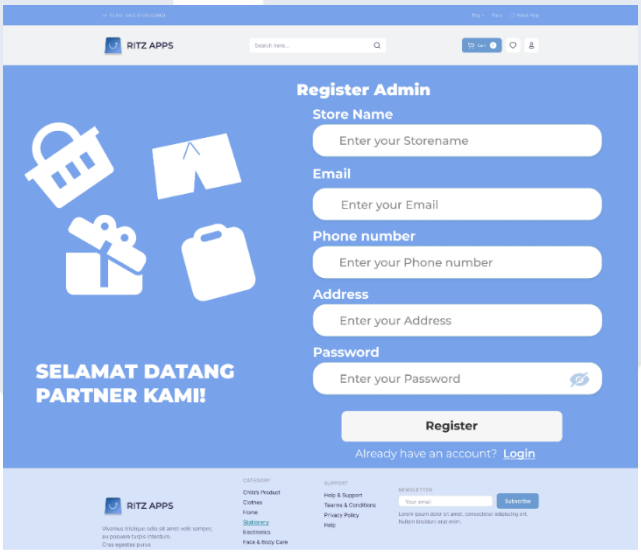
Gambar 3. 9 Struktur Database Website E-commerce

Pada kegiatan ini *developer* melakukan pembelajaran mengenai dokumen *project* yang kami sebut dengan dokumen SP. *Developer* juga mempelajari mengenai *database* yang telah ada dari batch sebelumnya. Tidak hanya itu, kami sebagai team juga membahas mengenai pengembangan *project* ini berikutnya dan membahas mengenai penambahan tabel ataupun kolom pada *database* yang diperlukan dan penambahan tabel ataupun kolom juga melakukan penambahan secara terus menerus selama kegaitan magang ini berlangsung. Gambar 3.9 adalah struktur *database* final yang telah diskusikan

3.2.1.3 Persiapan Mockup

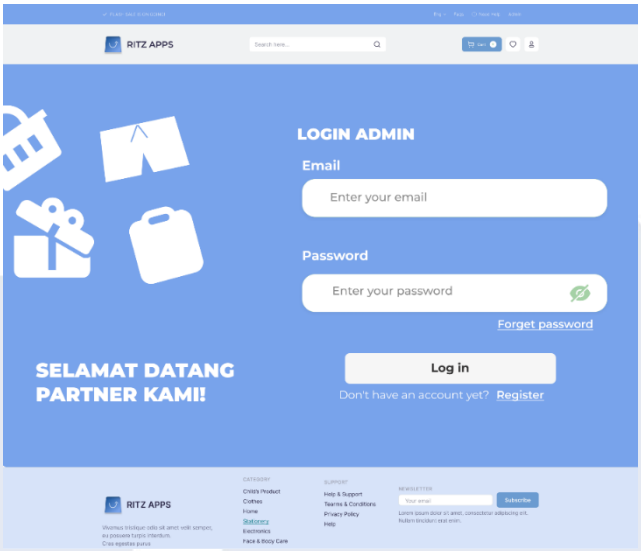
1) 8 Juli – 13 Juli

Pada kegiatan ini developer dan satu tim mempersiapkan tampilan mockup yang telah kami rancang pengembangan dari e-commerce. Developer melakukan persiapan mockup menggunakan figma sebagai desain mockup.



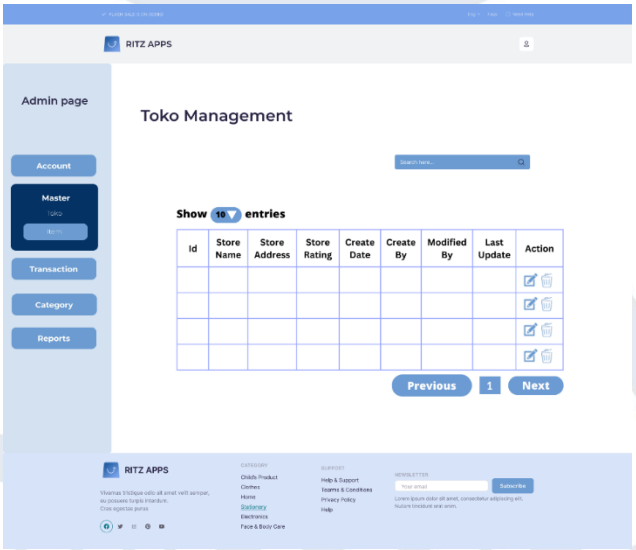
Gambar 3. 10 Mockup dari Register Admin

Persiapan mockup ini seperti pada gambar 3.10 mengenai register admin yang belum ada pada batch sebelumnya. Oleh karena itu, tim *developer* membuat mock-up untuk mempermudah *back-end developer* membuat tampilan register admin pada sisi *back-end*



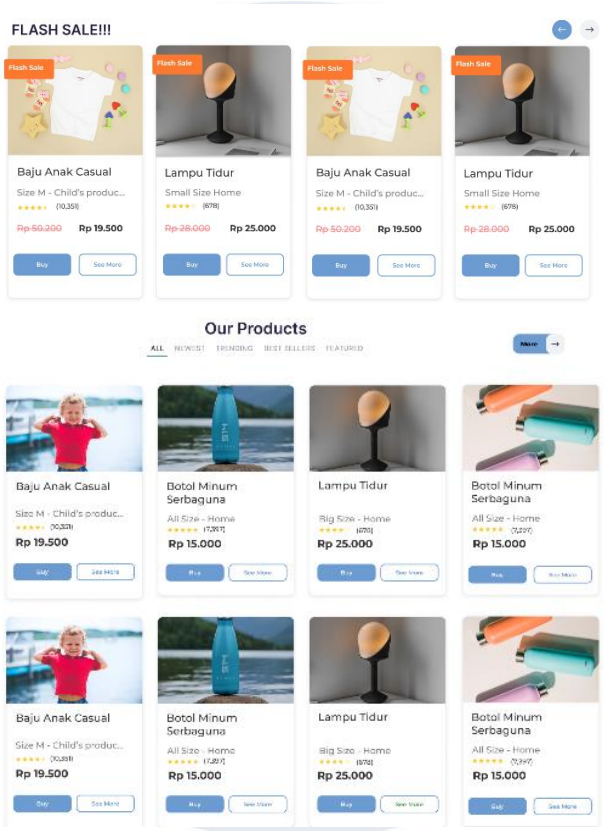
Gambar 3. 11 Mockup dari Login Admin

Selain itu juga ada penambahan mock-up seperti gambar 3.11 login admin. Penambahan mock-up ini untuk memudahkan *back-end developer* untuk mendapatkan gambaran mengenai tampilan login admin dan membantu *back-end developer* untuk *develop*



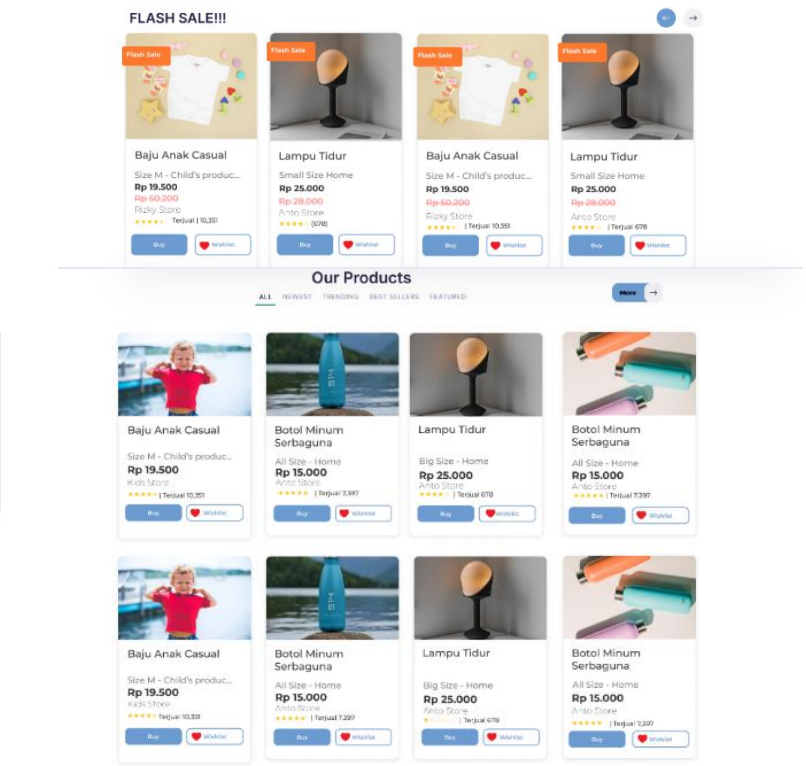
Gambar 3. 12 Contoh Mockup Toko Management

Tidak hanya pembuatan mockup pada sisi login admin dan register admin tapi membuat mockup untuk tampilan toko management. Pada gambar 3.12 ada penambahan mockup untuk toko management yang berisikan daftar nama-nama toko.



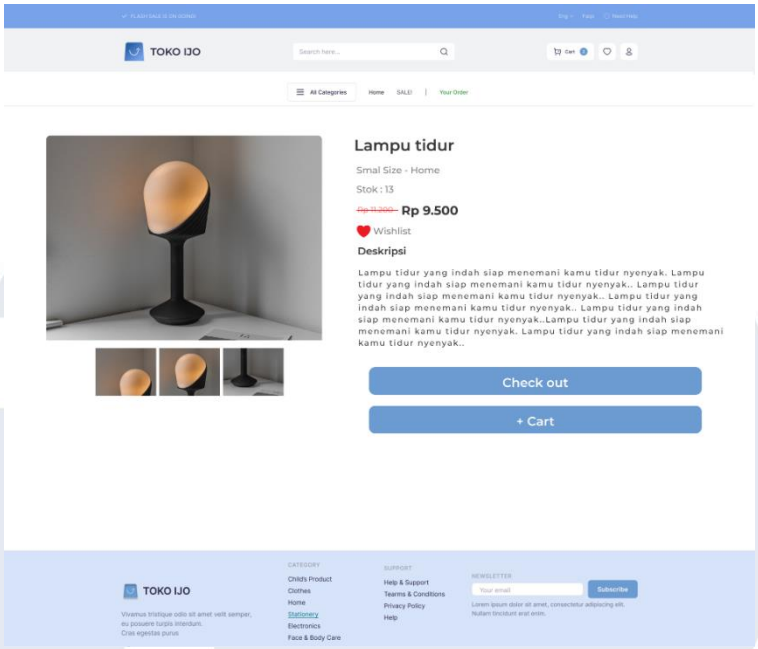
Gambar 3. 13 Contoh Mockup sebelum perubahan Home

Terakhir adalah mockup yang direncanakan untuk diubah. Pada gambar 3.13 pada card item yang bertulisan ‘See More’ akan diganti dengan tulisan “Wishlist” sehingga ini mempermudah user untuk memasukkan product yang mereka inginkan. Selain itu juga ada penambahan pada tampilan tersebut yaitu ada penambahan nama toko di card item.



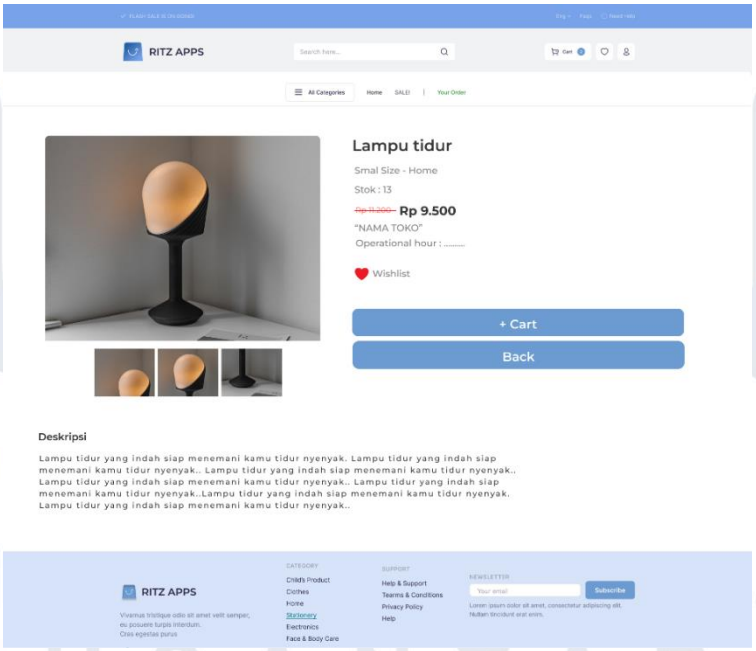
Gambar 3. 14 Mockup setelah perubahan Home

Gambar 3.14 merupakan hasil mock-up yang telah diubah dari kata "See More" menjadi kata "Wishlist". Selain itu juga telah adanya penambahan nama toko pada tampilan tersebut. Sementara untuk melihat detail produk maka bisa mengklik pada bagian gambar ataupun tulisan.



Gambar 3. 15 Mockup Detail Product batch pertama

Gambar 3.15 merupakan mockup detail product dari batch sebelumnya. Pada gambar 3.15 tidak ada nama toko sehingga pada batch sekarang ini memiliki tambahan yaitu nama toko dan waktu operasional toko.



Gambar 3. 16 Mockup perubahan Detail Product

Pada gambar 3.16 merupakan hasil mock-up yang memiliki perbedaan dengan *mockup* detail product di batch sebelumnya. Perbedannya terlihat pada saat ini detail product memiliki nama toko dan operational hour seperti gambar 3.16.

3.2.2 Pengembangan Back-end pada E-commerce secara Keseluruhan

Developer selama magang berlangsung diberikan tugas untuk mengembangkan sebuah *website e-commerce* yang telah ada pada batch sebelumnya. Pada batch sebelumnya hanya terdapat register dan login yang bisa digunakan 2 role yaitu admin dan user. Berikutnya pada batch sebelumnya tidak terdapat beberapa toko sehingga *e-commerce* tersebut ditujukan untuk 1 toko saja. Pada batch sebelumnya *website e-commerce* belum terdapat discount seperti birthday, event, member, payterm, point, return product. Oleh karena itu, pada batch sekarang ini diminta untuk membuat sebuah regsiter dan login bagi role admin dan memiliki tampilan mengenai toko management yang berisikan nama-nama toko yang bergabung ke dalam *website e-commerce* tersebut.

Selain itu, pada batch sekarang juga membuat sebuah discount event, member, birthday, payterm, point pada sisi transaction. Tidak hanya itu developer juga membuat user hanya bisa membeli pada 1 jenis toko saja yang ini berbeda dengan batch sebelumnya. Pada pengembangan *website* saat ini terdapat return product sehingga user bisa mengembalikan barang yang usak atau catat dengan maksimal 3 hari setelah menerima barang tersebut. Terakhir yaitu adanya perbaikan pada search bar yang pada batch sebelumnya tidak berfungsi dengan baik, lalu adanya perubahan tampilan homepage. Detail dari setiap pengembangan dan perbaikan fitur yang adad di *website e-commerce* akan dijelaskan secara detail sebagai berikut ini:

3.2.2.1 Pengembangan Admin

1) Pengembangan Admin Master Toko

a) 14 Juli -21 Juli

```php
public function index_headertk()
    {
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        $master = DB::table('users')->join('trole', 'users.id_role', '=', 'trole.id_role')
            ->get(['users.*', 'trole.vrole_name'])->where('id_role', 44441);
        return view('mastertoko', compact('master', 'nama_web'));
    }
```

Gambar 3. 17 Function menampilkan data Master Toko (Master Controller)

Pada kegiatan ini *developer* dan tim melakukan penambahan tampilan untuk master toko yang berisikan data user yang berlaku sebagai admin yang memiliki id_role 44441. Dalam pengerjaan menggunakan laravel ini tidak terlepas dari 3 bagian yaitu *controller* sebagai *function* code untuk menerima permintaan dari user, route untuk mengarahkan request ke *method* yang tepat dan view berisikan desain tampilan dari suatu web. *Developer* terlebih dahulu membuat *function* pada bagian master controller yang terlihat pada gambar 3.17. *Developer* membuat sebuah variabel dengan nama master yang berisikan data user yang memiliki id_role 44441 dengan query dari tabel user yang dijoinkan dengan table role dan mengambil id_role sebagai admin.

```php
//Master Toko
        Route::get('master/toko', 'App\Http\Controllers\MasterController@index_headertk');
        Route::post('master/toko/store', 'App\Http\Controllers\UserController@store_toko');
        Route::get('master/toko/edit/{id}', 'App\Http\Controllers\MasterController@index_updatetk');
        Route::post('master/toko/edit', 'App\Http\Controllers\MasterController@update_headertk');
        Route::get('master/toko/delete/{id}', 'App\Http\Controllers\MasterController@index_deletetk');
        Route::post('master/toko/delete', 'App\Http\Controllers\MasterController@delete_headertk');
```

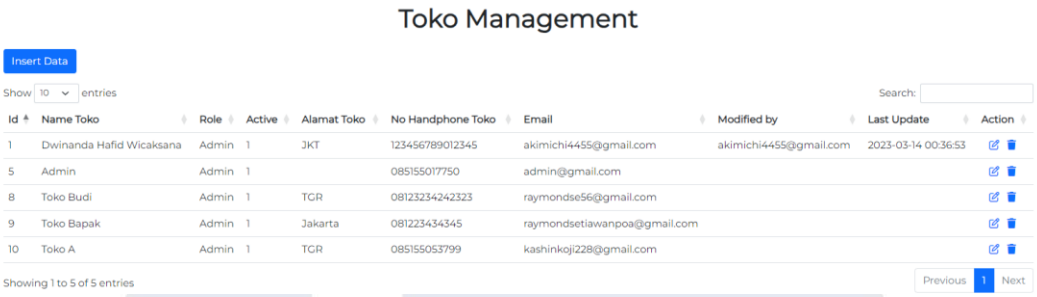
Gambar 3. 18 Route Master Toko

Setelah itu, melakukan *return view* ke tampilan mastertoko dengan mem-passing data ke tampilan mastertoko. *Developer* juga membuat sebuah *route* baru seperti gambar 3.18 agar *method* yang telah dibuat oleh *developer* bisa dibaca oleh view master toko. Gambar 3.18 memiliki beberapa *route* dengan *method* GET dan POST. *Method* GET ini berguna untuk mengambil data sementara post untuk post data. *Route* ini terdapat 6 route yaitu 3 *route* untuk GET dari *method* index untuk menampilkan data master toko ke tampilan website melalui *view* dan 3 *route method* POST untuk mengirimkan data yang datanya tidak tersimpan dalam URL seperti *insert*, *update* dan *delete* dari *function* yang telah dibuat.

```
<tbody>
            @foreach($master as $i => $u)
            <tr>
                <td>{{++$i}}</td>
                <td>{{$u->vname}}</td>
                <td>{{$u->vrole_name}}</td>
                <td>{{$u->istatus_user}}</td>
                <td>{{$u->vaddress}}</td>
                <td>{{$u->vno_telp}}</td>
                <td>{{$u->email}}</td>
                <td>{{$u->vmodi}}</td>
                {{-- @if ($u->vmodi == null)
                <td>-</td>
                @endif
                @if ($u->vmodi == 44441)
                <td>Admin</td>
                @endif
                @if ($u->vmodi == 44442)
                <td>Staff</td>
                @endif
                @if ($u->vmodi == 44443)
                <td>User</td>
                @endif

                @if ($u->dmodi == null)
                <td>-</td>
                @endif
                @if ($u->dmodi) --}}
                <td>{{$u->dmodi}}</td>
                {{-- @endif --}}
                    <td class="text-center">
                        <a href="/master/toko/edit/{{$u->id_user}}"><i class="fa-regular fa-pen-to-square me-2"></i></a>
                        <a href="/master/toko/delete/{{$u->id_user}}"><i class="fa-solid fa-trash me-2"></i></a>
                    </td>
                </tr>
            @endforeach
        </tbody>
```

Gambar 3. 19 Code tampilan Master Toko (Master_toko.blade.php)

Setelah membuat route, developer melakukan foreach pada variabel master untuk bisa menampilkan data-data yang memiliki id_role admin seperti gambar 3.19. Foreach dapat diartikan sebagai looping data dari tabel users.



Gambar 3. 20 Code tampilan Master Toko (Master_toko.blade.php)

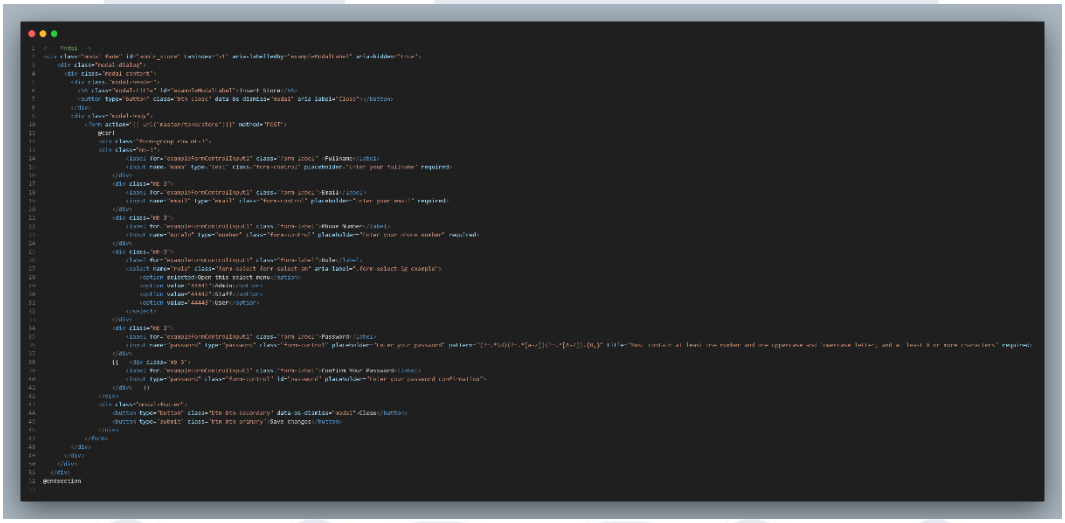
Hasil dari *foreach* dapat terlihat pada gambar 3.20. Hasil tersebut menampilkan semua data mengenai toko yang berada di database.

```php
public function store_toko(Request $request)
    {
        //check and prepare general data
        $picture = "blank_profilepicture.jpg";
        // $request->validate([
        //     'email' => 'required|unique:users',
        //     'password' => 'required',
        // ]);

        $data = $request->all(); // This will get all the request data.
        $userCount = User::where('email', $data['email']);
        if ($userCount->count()) {
            Alert::error('Error', 'That email address is already registered. You sure you don\'t have an account?');
            return redirect("/master/toko");
        } else {
            // insert data ke table users
            $check_insert = DB::table('users')->insert([
                'vname' => $request->nama,
                'email' => $request->email,
                'vno_telp' => $request->notelp,
                'password' => bcrypt($request->password),
                'istatus_user' => 1,
                'profile_picture' => $picture,
                'id_role' => $request->role,
                'vcrea' => Auth::user()->email,
                'dcrea' => Carbon::now()
            ]);

            if (!$check_insert) {
                Alert::error('Error', 'Data cannot be store');
            } else {
                Alert::Success('success', 'Data has been Store');
            }
        }
        return redirect('master/toko');
    }
```

Gambar 3. 21 Code untuk menambahkan data Toko

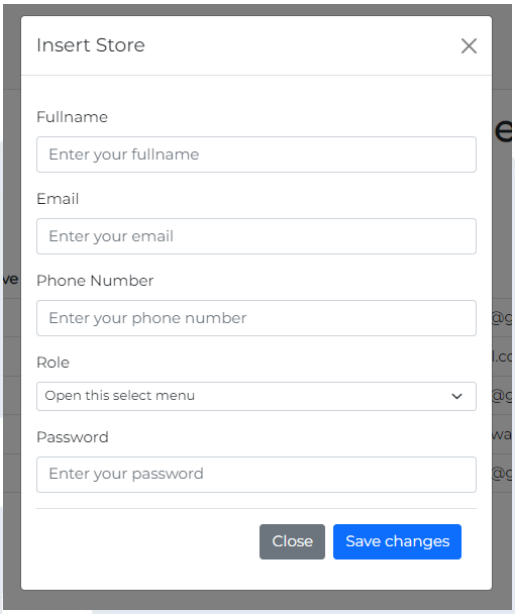
Selain menampilkan data master toko, *developer* juga melakukan *insert* toko dengan meletakan *method* ini di user controller dan diberi nama store_toko. Dalam melakukan *insert* toko, *developer* terlebih dahulu membuat variabel data yang berisikan semua request yang diinput oleh user dan terdapat variabel userCount yang berisikan query

dari tabel users dengan kondisi apakah email yang diinput users telah ada di tabel users. *Developer* juga menambahkan kondisi IF jika email telah ada maka akan muncul *alert* atau notifikasi. Jika kondisi email yang diinput tidak ada di database maka email tersebut bisa melakukan perintah DML (*Data Manipulation Language*) yaitu *insert* ke database users. Dalam melakukan insert juga terdapat kondisi jika *insert* tidak berhasil maka muncul *alert error'* tapi jika berhasil maka muncul *alert success* dan kembali ke tampilan master toko seperti terlihat pada gambar 3.21.



Gambar 3. 22 Modal untuk menambahkan data Toko (Master_toko.blade.php)

Gambar 3.22 ini merupakan modal untuk insert data toko yang diletakan dibawah code dari tampilan mastertoko. Dalam code insert terdapat name yang ini berhubungan dengan method store_toko dari request saat melakukan insert maka jika di modal name = password untuk password maka di method saat insert harus juga terdapat request-> password agar password dari modal bisa dimasukkan kedalam tabel users.



Gambar 3. 23 Modal untuk menambahkan data Toko (Master_toko.blade.php)

Hasil dari code modal ini dapat terlihat pada gambar 3.23. Hasil tersebut membuat user bisa menginput toko baru dengan mengisikan informasi mengenai nama, email, no telepon, role dan password.

```php
public function index_deletetk($id)
    {
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        $admin = DB::table('users')->where('id_user', $id)->first();
        return view('mastertoko_delete', compact('admin', 'nama_web'));
    }
```

Gambar 3. 24 Function menampilkan data Master Toko Delete (Master Controller)

Developer tidak hanya melakukan *insert* data master toko tapi *developer* juga melakukan penghapusan master toko. Pada gambar 3.24 merupakan *function* yang berguna untuk menampilkan data dari *request* yang ingin dihapus oleh user ke dalam variabel $id dengan nama *function* index_deletetk. Setelah itu, *developer* membuat sebuah variabel benama admin yang mengambil data dari table users dengan konsidi id_user dari request yang dipilih oleh user dengan mengunakan first()

untuk mengambil hasil pertama. Lalu mengirimkan variabel tersebut ke tampilan mastertoko_delete dengan mem-passing variabel admin.

```php
public function delete_headertk(Request $request)
    {
        $check = DB::table('users')->where(['id_user' => $request->id])->first();
        if (!$check) {
            Alert::error('Error', 'Data cannot be change');
        } else {
            DB::table('users')->where(['id_user' => $request->id])
                ->update([
                    'vname' => $request->name,
                    'istatus_user' => $request->status,
                    'vmodi' => Auth::user()->id_role,
                    'dmodi' => Carbon::now()
                ]);
            Alert::Success('success', 'Data has been Updated');
        }
        return redirect('master/toko');
    }

```

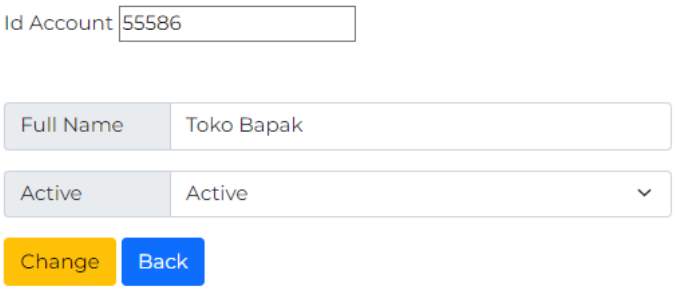
Gambar 3. 25 Function untuk delete data master toko

Setelah membuat index maka *developer* membuat sebuah *function* delete seperti pada gambar 3.25 yang memiliki inputan request dari user. *Developer* membuat sebuah variabel baru dengan nama check yang diambil dari tabel users dengan kondisi id_uer dari request->id yang berasal dari index. Jika terdapat id yang sesuai dengan request maka bisa melakukan *delete*. Dalam melakukan delete master toko ini tidak menggunakan perintah DML (*Data Manipulation Language*) *delete* tapi menggunakan update karena *developer* hanya mengupdate status user menjadi inactive. DML (*Data Manipulation Language*) *delete* tidak bisa digunakan karena tabel users memiliki join dengan tabel lainnya.

```
<!-- Menghubungkan dengan view template master -->
@extends('template')
@section('content')
<div class="card-body">
    <div class="container-fluid">
        <div class="row">
          <div class="col-md-4 ">
            <form action="{{ url('master/toko/delete')}}" method="post" enctype="multipart/form-data">
                @csrf
                <div class="form-group">
                <label for="">Id Account</label>
                    <input type="text" name="id" value="{{$admin->id_user}}">
                    <div class="input-group mb-3 mt-5">
                      <span class="input-group-text w-25" id="inputGroup-sizing-default"
                        >Full Name</span
                      >
                      <input
                        name="name"
                        type="text"
                        class="form-control"
                        value="{{$admin->vname}}"
                        aria-label="Sizing example input"
                        aria-describedby="inputGroup-sizing-default"
                      />
                    </div>
                    <div class="input-group mb-3">
                      <span class="input-group-text w-25" id="inputGroup-sizing-default"
                        >Active</span
                      >
                      <select name="status" class="form-select" aria-label="Default select example">
                        <option value="1" selected>Active</option>
                        <option value="0">Inactive</option>
                      </select>
                    </div>
                </div>
                <div class="form-group">
                    <button type="submit" class="btn btn-warning">Change</button>
                    <button type="button" onclick="history.back();" class="btn btn-primary">Back</button>
                </div>
            </form>
          </div>
        </div>
    </div>
</div>
@endsection
```

Gambar 3. 26 HTML dari tampilan mastertoko_delete

Gambar 3.26 adalah HTML dari mastertoko delete. Developer membuat sebuah form dengan method post yang ini akan berhubungan dengan route sebelumnya pada gambar 3.18, Selain itu, developer juga mengisi value dari variable admin seperti untuk full name valuenya berisikan {{$admin->vname}}. Agar bisa berpindah ke tampilan ini maka diperlukan sebuah input bernama id yang ini berasal dari function index $id yang berisikan value dari variabel admin dari id_user. Jika tidak menggunakan ini maka tampilan tidak akan ikut berpindah.



Gambar 3. 27 Tampilan delete master toko

Tampilan master toko delete dapat terlihat pada gambar 3.27. User selaku admin bisa mengactivekan atau non-activekan toko tersebut dengan memilih pada bagian active.

```
public function index_updatetk($id)
    {
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        //$category = DB::table('tcategory')->select('tcategory.id_category','tcategory.vcategory')->get();
        $master = DB::table('users')->where('id_user', $id)->first();
        return view('mastertoko_edit', compact('master', 'nama_web'));
    }
```

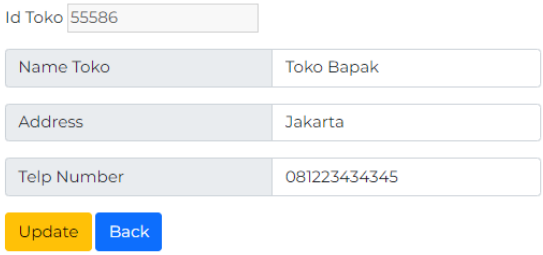
Gambar 3. 28 Function menampilkan data untuk mengubah master toko

Pada gambar 3.28 merupakan function untuk menampilkan data saat ingin mengubah data dari request yang ingin dihapus oleh user kedalam variabel $id dengan nama *function* index_updatetk. Hal ini sama seperti delete data di master toko. *Developer* membuat sebuah variabel benama master yang mengambil data dari table users dengan konsidi id_user dari request yang dipilih oleh user dengan mengunakan first() untuk mengambil hasil pertama. Lalu mengirimkan variabel tersebut ke tampilan mastertoko_edit dengan mem-passing variabel master.

```php
public function update_headertk(Request $request)
    {
        $query = DB::table('users')->where('id_user', $request->id)->update([
            'vname' => $request->name,
            'id_user' => $request->id,
            'vno_telp' => $request->no_telp,
            'vaddress' => $request->address,
            'vcrea' => Auth::user()->email,
            'dcrea' => Carbon::now()
        ]);

        if (!$query) {
            Alert::error('Error', 'Data cannot be update');
        } else {
            Alert::Success('success', 'Data has been updated');
        }

        return redirect('master/toko');
    }

```

Gambar 3. 29 Function untuk mengubah data master toko

Setelah membuat *index* untuk *update* toko maka perlu sebuah function untuk bisa mnegubah data dari tabel users. *Developer* membuat sebuah *variable* query yang berisikan perintah DML (*Data Manipulation Language*) untuk *update* dari table users dengan kondisi id_users sesuai dengan parameter request id yang dipilih oleh users. Hal yang akan terupdate ini berupa nama, no telp, dan alamat dapat terlihat pada gambar 3.29. Jika berhasil dalam mengubah data maka akan muncul sebuah *alert success* tapi jika tidak berhasil maka akan muncul sebuah *alert error*.



Gambar 3. 30 Tampilan saat edit data master toko

Gambar 3.30 merupakan hasil dari tampilan edit pada master toko yang membuat admin bisa mengubah informasi toko tersebut. HTML dari tampilan tersebut terdapat pada gambar 3.31.

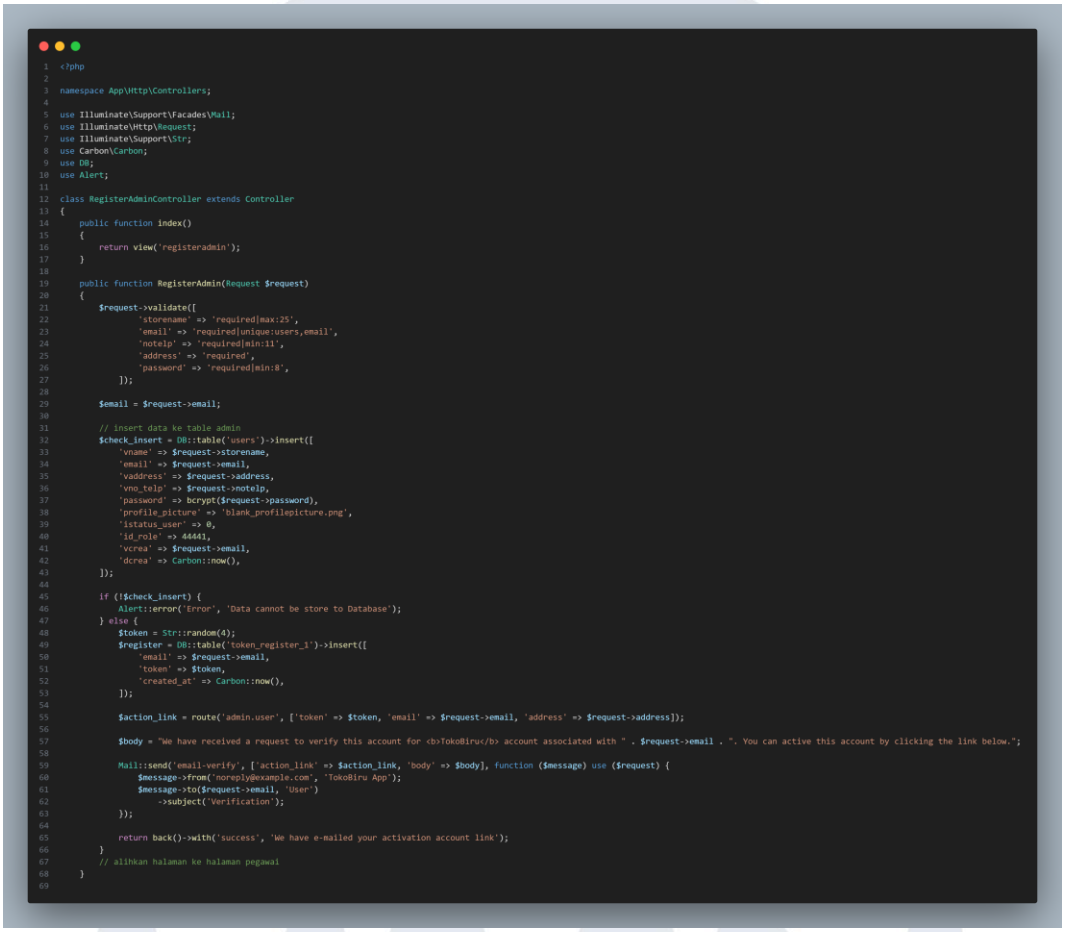
```
<!-- Menghubungkan dengan view template master -->
@extends('template')
@section('content')
<div class="card-body">
    <div class="container-fluid">
        <div class="row">
          <div class="col-md-4 ">
            <form action="{{ url('master/toko/edit')}}" method="post" enctype="multipart/form-data">
                @csrf
                <div class="form-group">
                    <label for="">Id Toko</label>
                    <input type="text" name="id" value="{{$master->id_user}}" disabled>
                    <input type="text" name="id" value="{{$master->id_user}}" hidden>
                    <div class="input-group mb-3 mt-3">
                        <span class="input-group-text w-50" id="inputGroup-sizing-default"
                          >Name Toko</span
                        >
                        <input
                          name="name"
                          type="text"
                          class="form-control"
                          value="{{$master->vname}}"
                          aria-label="Sizing example input"
                          aria-describedby="inputGroup-sizing-default"
                          required
                        />
                    </div>
                    <div class="input-group mb-3 mt-3">
                        <span class="input-group-text w-50" id="inputGroup-sizing-default"
                          >Address</span
                        >
                        <input
                          name="address"
                          type="text"
                          class="form-control"
                          maxlength= "100"
                          value="{{$master->vaddress}}"
                          aria-label="Sizing example input"
                          aria-describedby="inputGroup-sizing-default"
                          required
                        />
                    </div>
```

Gambar 3. 31 HTML dari mastertoko_edit

Gambar 3.31 merupakan HTML dari mastetoko_edit, *developer* melakukan hal yang sama seperti delete master toko. *Developer* melakukan pengisi value dari tampilan yang telah dibuat oleh *front end developer*. *Developer* juga tidak lupa menambahkan input bernama id yang ini berasal dari function index_updatetk $id yang berisikan value dari variabel master dari id_user dan menambahkan route seperti gambar 3.18.

2) Pengembangan Register dan Login Admin

a) 24 Juli – 29 Juli



```php
<?php

namespace App\Http\Controllers;

use Illuminate\Support\Facades\Mail;
use Illuminate\Http\Request;
use Illuminate\Support\Str;
use Carbon\Carbon;
use DB;
use Alert;

class RegisterAdminController extends Controller
{
    public function index()
    {
        return view('registeradmin');
    }

    public function RegisterAdmin(Request $request)
    {
        $request->validate([
                'storename' => 'required|max:25',
                'email' => 'required|unique:users,email',
                'notelp' => 'required|min:11',
                'address' => 'required',
                'password' => 'required|min:8',
            ]);

        $email = $request->email;

        // insert data ke table admin
        $check_insert = DB::table('users')->insert([
            'vname' => $request->storename,
            'email' => $request->email,
            'vaddress' => $request->address,
            'vno_telp' => $request->notelp,
            'password' => bcrypt($request->password),
            'profile_picture' => 'blank_profilepicture.png',
            'istatus_user' => 0,
            'id_role' => 44441,
            'vcrea' => $request->email,
            'dcrea' => Carbon::now(),
        ]);

        if (!$check_insert) {
            Alert::error('Error', 'Data cannot be store to Database');
        } else {
            $token = Str::random(4);
            $register = DB::table('token_register_1')->insert([
                'email' => $request->email,
                'token' => $token,
                'created_at' => Carbon::now(),
            ]);

            $action_link = route('admin.user', ['token' => $token, 'email' => $request->email, 'address' => $request->address]);

            $body = "We have received a request to verify this account for <b>TokoBiru</b> account associated with " . $request->email . ". You can active this account by clicking the link below.";

            Mail::send('email-verify', ['action_link' => $action_link, 'body' => $body], function ($message) use ($request) {
                $message->from('noreply@example.com', 'TokoBiru App');
                $message->to($request->email, 'User')
                        ->subject('Verification');
            });

            return back()->with('success', 'We have e-mailed your activation account link');
        }
        // alihkan halaman ke halaman pegawai
    }

```

Gambar 3. 32 Function Register Admin

Pada kegiatan ini, *developer* melakukan pengembangan untuk melakukan register admin dan membuat login admin. Gambar 3.32 merupakan function dari register admin. *Developer* terlebih dahulu membuat function index yang berisikan return view register admin. Setelah itu *developer* menvalidasi inputan user dengan *method* validate dari nama toko, email, no telp, alamat, dan password. Lalu membuat sebuah variabel dengan nama check_insert yang menggunakan perintah DML (*Data Manipulation Language*) insert untuk memasukan data kedalam tabel users. Jika berhasil dalam melakukan inputan ke dalam tabel user maka akan mengirimkan token dari variabel token dan

developer menambahkan variabel register yang berisikan perintah DML insert kedalam tabel token_register_1 yang berisikan token dan alamat email yang telah diinput oleh user.

```
1  //Register admin
2  Route::get('registeradmin', $ctrl.'\RegisterAdminController@index')->name('verified.admin_account.form');
3  Route::post('registeradmin', $ctrl.'\RegisterAdminController@RegisterAdmin')->name('verified.admin_account.link');
4
```

Gambar 3. 33 Route dari Register Admin

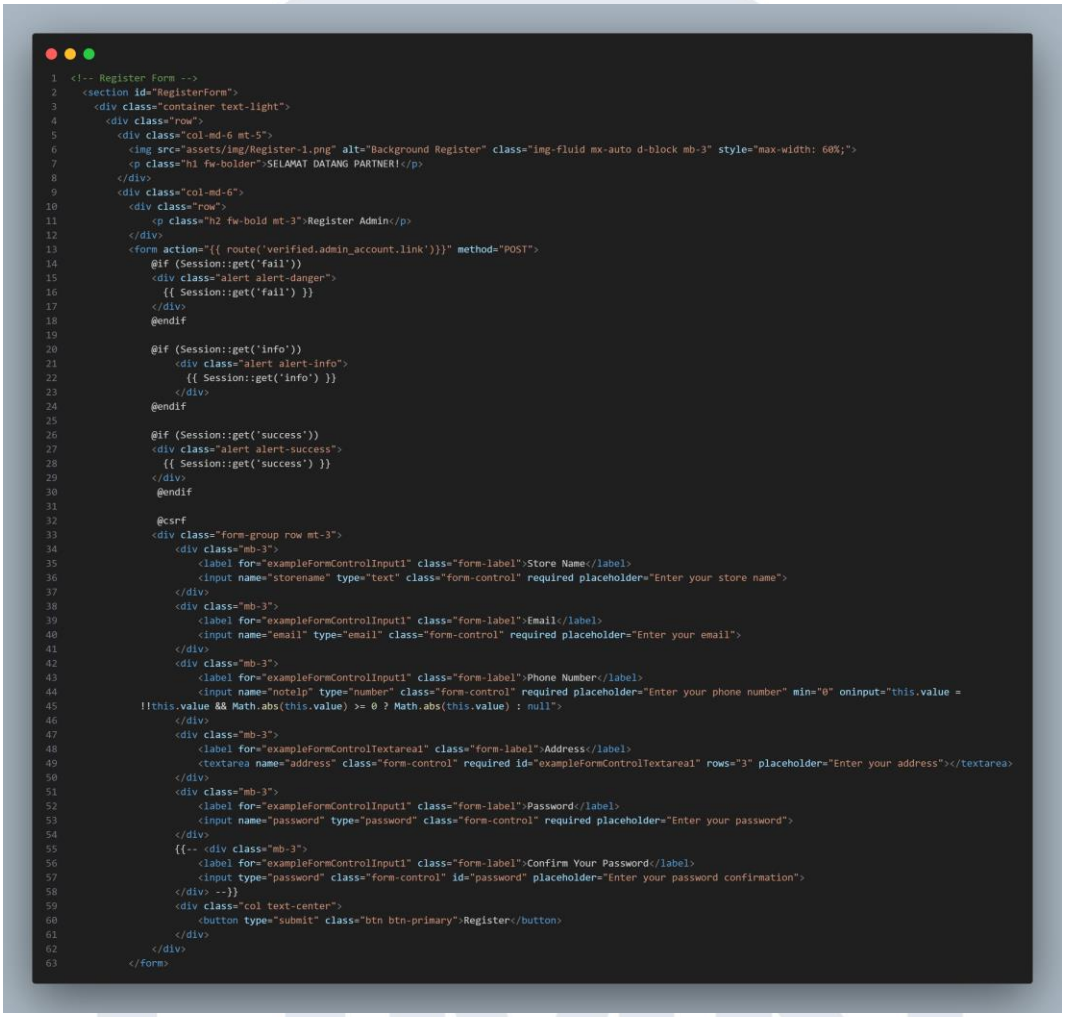
Developer juga membuat route seperti gambar 3.33. Route tersebut berguna untuk meng-*route* request yang diberikan oleh user sehingga bisa tepat sasaran dari permintaan user.

```
1  MAIL_MAILER=smtp
2  MAIL_HOST=smtp.gmail.com
3  MAIL_PORT=587
4  MAIL_USERNAME=tokobiru191@gmail.com
5  MAIL_PASSWORD=owysafijyprfehou
6  MAIL_ENCRYPTION=tls
7  MAIL_FROM_ADDRESS=tokobiru191@gmail.com
8  MAIL_FROM_NAME="${APP_NAME}"
```

Gambar 3. 34 Variable env untuk Email Verification

Lalu akan dibuat sebuah generate link untuk user melakukan verifikasi akun pada email user. Cara untuk mengirimkan *email verification* ini sama dengan melakukan register user yang terjadi pada bacth pertama. Agar bisa melakukan verifikasi email maka terlebih dahulu *developer* mengisi value mail dari file .env seperti pada gambar 3.34. Dalam melakukan pengiriman email menggunakan *Simple Mail Transfer Protocol* (SMTP) dengan nomor port 587. Mail_username ini menggunakan email tokobiru191@gmail.com yang telah dibuat oleh

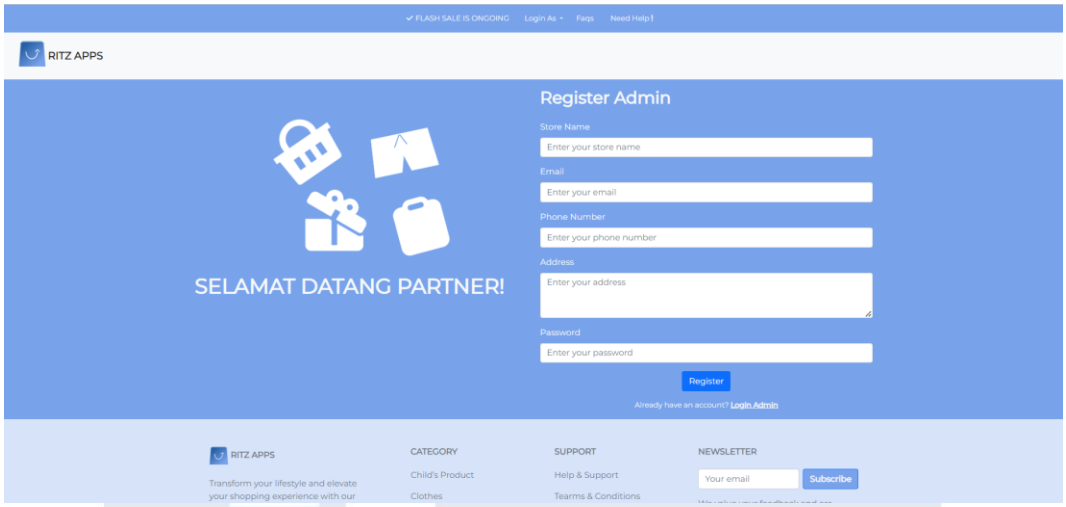
batch pertama dan mail passwrod ini disesuaikan dengan app pasword yang telah terencrypt oleh google.



```
<!-- Register Form -->
<section id="RegisterForm">
  <div class="container text-light">
    <div class="row">
      <div class="col-md-6 mt-5">
        <img src="assets/img/Register-1.png" alt="Background Register" class="img-fluid mx-auto d-block mb-3" style="max-width: 60%;">
        <p class="h1 fw-bolder">SELAMAT DATANG PARTNER!</p>
      </div>
      <div class="col-md-6">
        <div class="row">
            <p class="h2 fw-bold mt-3">Register Admin</p>
        </div>
        <form action="{{ route('verified.admin_account.link')}}" method="POST">
           @if (Session::get('fail'))
           <div class="alert alert-danger">
             {{ Session::get('fail') }}
           </div>
           @endif

           @if (Session::get('info'))
                <div class="alert alert-info">
                  {{ Session::get('info') }}
                </div>
           @endif

           @if (Session::get('success'))
           <div class="alert alert-success">
             {{ Session::get('success') }}
           </div>
            @endif

            @csrf
           <div class="form-group row mt-3">
                <div class="mb-3">
                    <label for="exampleFormControlInput1" class="form-label">Store Name</label>
                    <input name="storename" type="text" class="form-control" required placeholder="Enter your store name">
                </div>
                <div class="mb-3">
                    <label for="exampleFormControlInput1" class="form-label">Email</label>
                    <input name="email" type="email" class="form-control" required placeholder="Enter your email">
                </div>
                <div class="mb-3">
                    <label for="exampleFormControlInput1" class="form-label">Phone Number</label>
                    <input name="notelp" type="number" class="form-control" required placeholder="Enter your phone number" min="0" oninput="this.value =
          !!this.value && Math.abs(this.value) >= 0 ? Math.abs(this.value) : null">
                </div>
                <div class="mb-3">
                    <label for="exampleFormControlTextarea1" class="form-label">Address</label>
                    <textarea name="address" class="form-control" required id="exampleFormControlTextarea1" rows="3" placeholder="Enter your address"></textarea>
                </div>
                <div class="mb-3">
                    <label for="exampleFormControlInput1" class="form-label">Password</label>
                    <input name="password" type="password" class="form-control" required placeholder="Enter your password">
                </div>
                {{-- <div class="mb-3">
                    <label for="exampleFormControlInput1" class="form-label">Confirm Your Password</label>
                    <input type="password" class="form-control" id="password" placeholder="Enter your password confirmation">
                </div> --}}
                <div class="col text-center">
                    <button type="submit" class="btn btn-primary">Register</button>
                </div>
            </div>
        </form>
```

Gambar 3. 35 HTML dari Register Admin

Developer juga menambahkan route dari HTML register admin pada bagian *form* seperti pada gambar 3.35 agar tampilan HTML ini bisa ditampilkan dan bisa berfungsi dengan baik. *Developer* juga mengisi name pada input sesuai dengan *validate* yang telah dilakukan oleh *developer* di *function* register admin seperti pada gambar 3.36.



Gambar 3. 36 Tampilan Register Admin

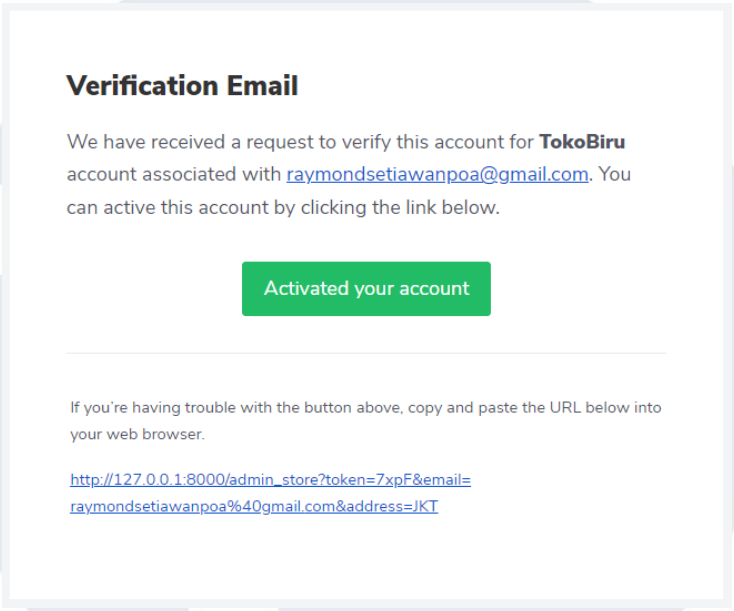
Pada gambar 3.36 merupakan hasil tampilan dari register admin yang user menginput data-data yang diminta pada tampilan register admin.

```php
public function RegisterLinkAdmin(Request $request, $token = null)
    {

        $user = DB::table('users')->where(['email' => $request->email])->first();
        $check_token = DB::table('token_register_1')->where([
            'email' => $request->email,
            'token' => $request->token,
        ])->first();


        if (!$check_token) {
            return back()->withInput()->with('fail', 'Invalid token');
        } else {
            DB::table('taddress')->insert([
                'vreceiver_name' => $user->vname,
                'vaddress' => $request->address,
                'vcrea' => $request->email,
                'dcrea' => Carbon::now(),
                'id_user' => $user->id_user,
                'istatus_address' => 1,
            ]);
            DB::table('users')->where(['email' => $request->email])->update(['istatus_user' => 1]);
            DB::table('token_register_1')->where([
                'email' => $request->email
            ])->delete();

            return redirect('loginadmin')->with('info', 'Your account has been activated, you can login to our website')->with('verifiedEmail', $request->email);
        }
    }
```

Gambar 3. 37 Function Register Link Admin

Setelah itu, *Developer* membuat function RegisterLinkAdmin seperti pada gambar 3.37 agar saat user melakukan verifikasi email akan dilacak dengan function ini yang nanti akan mengambil data dari tabel token_register_1. Jika token tidak benar maka akan muncul invalid token tapi jika berhasil maka akan melakukan insert data ke dalam tabel

address dan mengubah data istatus_user dari tabel users menjadi 1 yaitu active serta pada data tabel token_register_1 akan dihapus.



Gambar 3. 38 Tampilan Verification Email

User akan mendapatkan sebuah email mengenai active account dari system toko biru. Ketika pengguna mengklik “Actived your account” seperti pada gambar 3.38 maka akan langung ke tampilan login admin. Dengan berpindah ke halaman login admin maka user telah berhasil untuk register dan dapat secara langsung melakukan login.

```php
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use DB;
use Alert;
use App\Models\User;

class LoginAdminController extends Controller
{


    public function index()
    {
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        return view('loginadmin', compact('nama_web'));
    }

    public function authenticateAdmin(Request $request)
    {
        request()->validate([
                'email' => 'required',
                'password' => 'required',
            ]);
        $kredensil = $request->only('email', 'password');

        // $user=User::where('username',$request->username)->first();
        if (Auth::attempt($kredensil)) {
            // Auth::guard('web')->login($user);
            $user = Auth::user();
            if ($user->istatus_user == 1) {
                if ($user->id_role == 44441) {
                    return redirect()->intended('admin');
                } elseif ($user->id_role == 44442) {
                    return redirect()->intended('staff');
                }
            } elseif ($user->istatus_user == 0) {
            }
            Alert::error('Error', 'Your account not active yet, please check your email for activation');
            return back();
        }
        Alert::error('Error', 'Invalid Email or Password');
        return back();
    }
```

Gambar 3. 39 Function Login Admin (LoginAdminController.php)

Setelah berhasil membuat sebuah register admin maka berikutnya *developer* membuat login untuk admin. Pertama *developer* membuat sebuah function index untuk menampilkan tampilan login admin seperti pada gambar 3.39. Setelah itu, *developer* membuat function authenticateAdmin. Function ini berguna untuk memeriksa inputan user saat melakukan login apakah sesuai dengan data yang didalam tabel users dan id_role dari user tersebut. Jika memiliki id_role sebagai admin maka akan berhasil masuk.

```php
Route::post('proses_login_admin', $ctrl.'\LoginAdminController@authenticateAdmin');
```

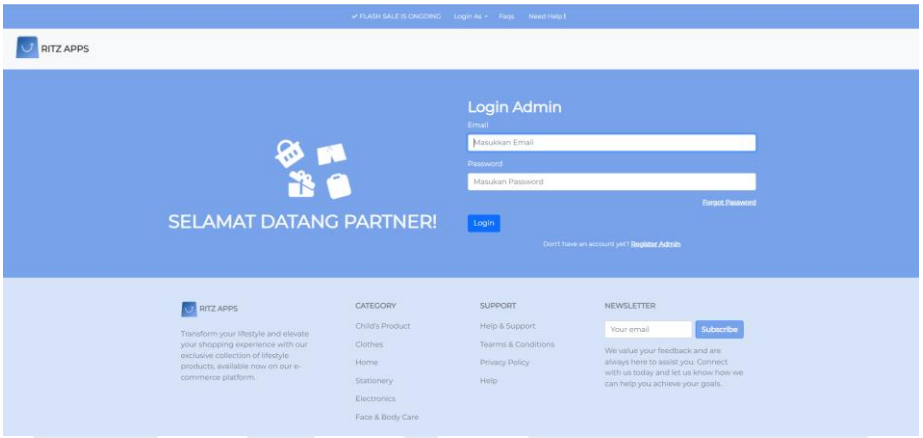
Gambar 3. 40 Route dalam proses login admin

Sebelum itu terdapat route yang menghubungan *function* tersebut dengan tampilan form login admin seperti pada gambar 3.40.

```blade
<!-- Login Form -->
<section id="LoginForm">
  <div class="container text-light">
    <div class="row align-items-center pt-5 pb-5">
      <div class="col-md-6 mt-5">
        <img src="assets/img/Register-1.png" alt="Background Register" class="img-fluid mx-auto d-block mb-3" style="max-width: 30%;">
        <p class="h1 fw-bolder">SELAMAT DATANG PARTNER!</p>
      </div>

      <div class="col-md-6">
        <div class="row">
        <p class="h2 fw-bold mt-3">Login Admin</p>
        </div>
        <form method="POST" action="{{ url('proses_login_admin')}}">
          @if (Session::get('info'))
            <div class="alert alert-info">
              {{ Session::get('info') }}
            </div>
          @endif
          @csrf
          <div class="mb-3">
            <label for="exampleFormControlInput1" class="form-label">Email</label>
            <input id="email" type="text" class="form-control " name="email" value="{{ old('email') }}" placeholder="Masukkan Email" required autocomplete="email" autofocus>
          </div>

          <div class="mb-3">
            <label for="exampleFormControlInput1" class="form-label">Password</label>
            <input id="password" type="password" class="form-control " name="password" placeholder="Masukan Password" required autocomplete="current-password">
          </div>
          <div class="row mt-3">
            <p class="small text-end"><a href="forgot_password" class="text-light fw-bold" >Forgot Password</a></p>
          </div>

          <button type="submit" class="btn btn-primary btn-user btn-block">
              Login
          </button>
        </form>
        <div class="row mt-3">
          <p class="small text-center">Don't have an account yet? <a href="registeradmin" class="text-light fw-bold" >Register Admin</a></p>
        </div>
      </div>
```

Gambar 3. 41 HTML untuk Login Admin

Gambar 3.41 merupakan HTML dari login admin, *developer* menambahkan alamat route pada bagian form sehingga tampilan login admin bisa muncul di web browser pengguna



Gambar 3. 42 Tampilan Login Admin

Pada gambar 3.42 merupakan hasil dari tampilan login dan user telah bisa melakukan login ketika akun usernya telah *active* dari email yang didapatkan.

```php
public function dashboard_admin(Request $request)
    {
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        $user = User::join('trole', 'users.id_role', '=', 'trole.id_role')
            ->get(['users.*', 'trole.vrole_name'])->where('id_role', 44443);
        return view('admin', compact('user', 'nama_web'));
    }
```

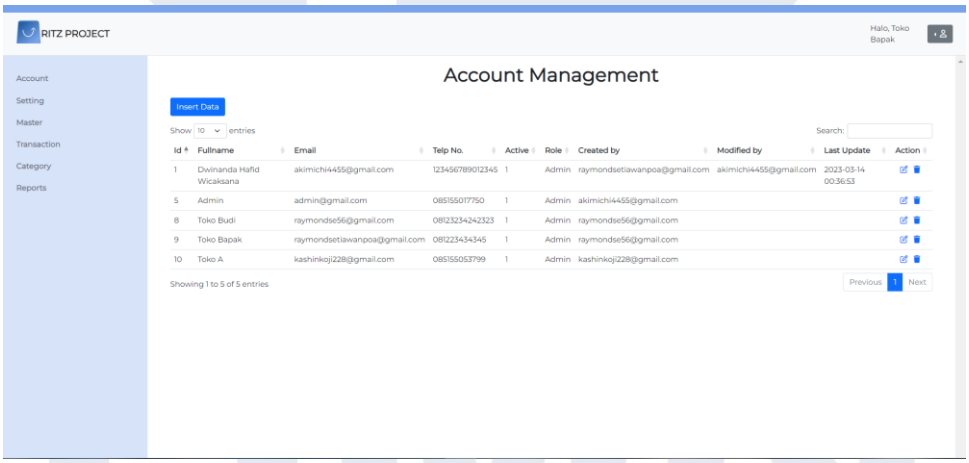
Gambar 3. 43 Function setelah berhasil login admin

Gambar 3.43 adalah function setelah pengguna berhasil login maka pengguna akan menuju tampilan admin yang berisikan data-data dan pengaturan mengenai *e-commerce*.

```php
// router admin
Route::group(['middleware' => ['auth']], function () {
    Route::group(['middleware' => ['cek_login:admin']], function () {

        Route::get('admin', 'App\Http\Controllers\LoginAdminController@dashboard_admin');
        Route::get('admin/edit/{id}','App\Http\Controllers\UserController@edit');
        Route::post('admin/edit','App\Http\Controllers\UserController@update');
        Route::get('admin/delete/{id}','App\Http\Controllers\UserController@delete');
        Route::post('admin/store','App\Http\Controllers\UserController@store');
        Route::post('account_delete', 'App\Http\Controllers\UserController@update_delete');
```

Gambar 3. 44 Route group untuk id_role admin

Sementara gambar 3.44 merupakan *route group* yang memiliki id_role sebagi admin Sehingga semua function yang berhubungan dengan admin akan diletakan di route tersebut menjadi satu kesatuan. Begitu juga dengan user ketika memiliki id_role user maka *function* yang berhubungan dengan user akan digabung menjadi 1 route group.



Gambar 3. 45 Tampilan setelah berhasil login

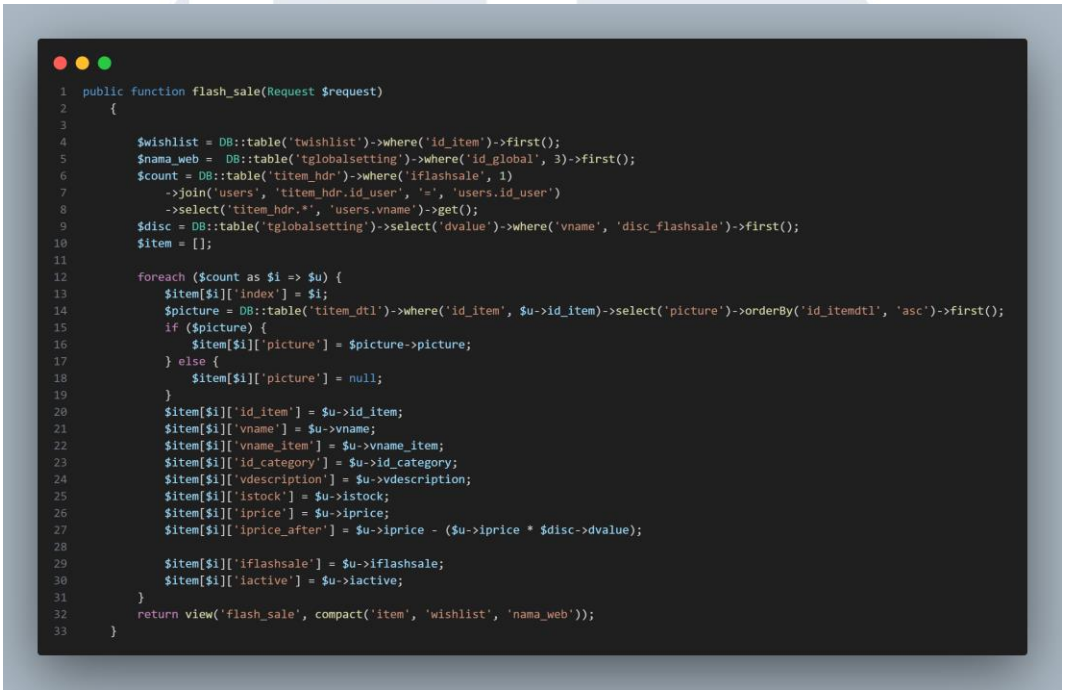
Pada gambar 3.45 merupakan tampilan ketika user telah berhasil login pada sisi admin sehingga user bisa mengatur *website e-commerce*.

3.2.2.2 Pengembangan Transaction

1) Bug Fixing Fitur Wishlist dan See More

a) 31 Juli – 3 Agustus

Pada kegiatan hari ini *developer* didapatkan tugas untuk memperbaiki fitur *wishlist*, *see more* dan menampilkan nama toko pada tampilan flash_sale, product more, dan home yang berisikan *product-product* toko. *Developer* memperbaiki dari batch sebelumnya.



```php
public function flash_sale(Request $request)
    {

        $wishlist = DB::table('twishlist')->where('id_item')->first();
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        $count = DB::table('titem_hdr')->where('iflashsale', 1)
            ->join('users', 'titem_hdr.id_user', '=', 'users.id_user')
            ->select('titem_hdr.*', 'users.vname')->get();
        $disc = DB::table('tglobalsetting')->select('dvalue')->where('vname', 'disc_flashsale')->first();
        $item = [];

        foreach ($count as $i => $u) {
            $item[$i]['index'] = $i;
            $picture = DB::table('titem_dtl')->where('id_item', $u->id_item)->select('picture')->orderBy('id_itemdtl', 'asc')->first();
            if ($picture) {
                $item[$i]['picture'] = $picture->picture;
            } else {
                $item[$i]['picture'] = null;
            }
            $item[$i]['id_item'] = $u->id_item;
            $item[$i]['vname'] = $u->vname;
            $item[$i]['vname_item'] = $u->vname_item;
            $item[$i]['id_category'] = $u->id_category;
            $item[$i]['vdescription'] = $u->vdescription;
            $item[$i]['istock'] = $u->istock;
            $item[$i]['iprice'] = $u->iprice;
            $item[$i]['iprice_after'] = $u->iprice - ($u->iprice * $disc->dvalue);

            $item[$i]['iflashsale'] = $u->iflashsale;
            $item[$i]['iactive'] = $u->iactive;
        }
        return view('flash_sale', compact('item', 'wishlist', 'nama_web'));
    }
```

Gambar 3. 46 Function Flash sale *(UserPageController.php)*

Developer melakukan penambahan variabel *wishlist* yang memanggil tabel wishlist dengan kondisi id_item. Variabel ini digunakan untuk melakukan *wishlist*. Setelah itu, agar bisa menambahkan product di flash sale maka perlu variabel count yang mengambil data dari tabel item_hdr berisikan item dengan kondisi iflashsale = 1 dan perlu melakukan join dengan tabel users agar bisa mengambil nama toko dengan *method* GET. Dalam menggunakan *method* GET maka diperlukan looping dengan *foreach* agar data dapat dibaca sesuai index baris seperti pada gambar 3.46. Setelah itu akan mempassing data item, wishlist tersebut ke tampilan flash sale. Agar data tersebut bisa digunakan di tampilan flash_sale.

```
@foreach ($item as $i => $u)
<div class="col-md-3 mb-3">
  <div class="card w-100 shadow" style="width: 18rem; height: 100%">
    @if ($u['picture'] != null)
    <a href="{{url('product-info').'/'.$u['id_item']}}">
      <img src="{{asset('assets/picture').'/'.($u['picture'])}}" class="card-img-top" alt="..." style="width:100%; height:250px" />
    </a>
    @endif
    @if ($u['picture'] == null)
    <a href="{{url('product-info').'/'.$u['id_item']}}">
      <img src="{{asset('assets/picture/blank_profilepicture.png')}}" class="card-img-top" alt="..." style="width:100%; height:250px" />
    </a>
    @endif
    <div class="card-body">
      <h5 class="card-title fw-bold"><a href="{{url('product-info').'/'.$u['id_item']}}">{{$u['vname_item']}}</a></h5>
      <p class="text-muted card-text " style="font-size: 1rem">
        {{$u['vdescription']}}
      </p>
      <p class="text-muted card-text " style="font-size: 1rem">
        {{$u['vname']}}
      </p>
      <img src="{{asset('assets/img/Review Flash Sale-01.png')}}" alt="" />
```

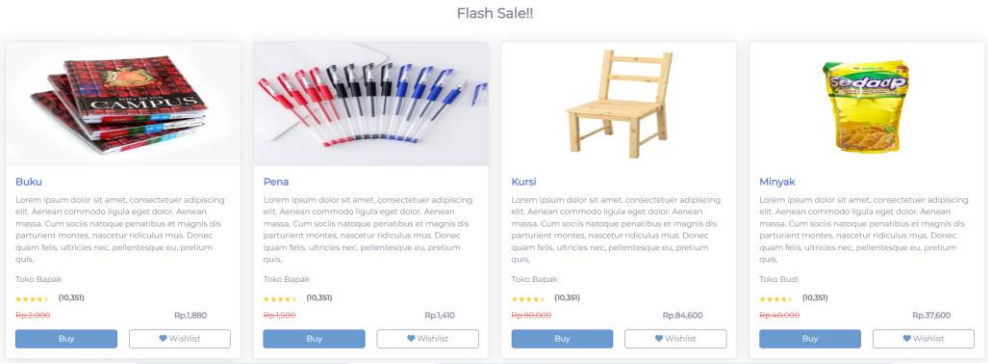
Gambar 3. 47 HTML Flash Sale untuk Detail product

Berikutnya di HTML flash_sale perlu melakukan *foreach* dari data yang di passing dari *function* flash_sale seperti pada gambar 3.47. Hal ini diperlukan agar data yang bersifat array ini bisa ditampilkan di *web browser*. Selain itu, *developer* juga menambahkan *hypertext reference* untuk detail product pada bagian gambar dan nama item sehingga ketika user mengklik bagian foto atau product akan berpindah ke detail product serta menambahkan nama toko.

```
<div class="col-md text-end">
  @if (is_null($wishlist))
  <a href="/wishlist/add/{{$u['id_item']}}">
    <button type="button" class="btn btn-outline-secondary d-bloxk w-100">
      <i class="fa-solid fa-heart" style="color: #6b9bd0"></i>
      Wishlist
    </button>
  </a>
  @endif
</div>
<div class="text-muted small">
  @if (!is_null($wishlist))
  <a href="/wishlist/add/{{$u['id_item']}}">
    <button type="button" class="btn btn-outline-secondary d-bloxk w-100">
      <i class="fa-solid fa-heart" style="color: red"></i> Wishlist
    </button>
  </a>
  @endif
</div>
```

Gambar 3. 48 HTML Flash Sale untuk wishlist

Tidak hanya itu, *developer* juga menambahkan kondisi seperti pada gambar 3.48 ketika tabel wishlist kosong maka wishlist tetap berwana biru tapi ketika tabel wishlist tidak kosong maka berubah warna.



Gambar 3. 49 Tampilan Flash sale yang telah berubah

Hasil dari perubahan ini seperti pada gambar 3.49. Dari hasil tersebut bahwa telah terdapat wihlist dan memudahkan user untuk menyimpan product tersebut kedalam menu wishlist.

```php
public function product_more()
    {
        $wishlist = DB::table('twishlist')->where('id_item')->first();
        $nama_web = DB::table('tglobalsetting')->where('id_global', 3)->first();
        $count = DB::table('titem_hdr')->where('iflashsale', 0)
            ->join('users', 'titem_hdr.id_user', '=', 'users.id_user')
            ->select('titem_hdr.*', 'users.vname')->get();
        $category = DB::table('tcategory')->get();
        $item = [];

        foreach ($count as $i => $u) {
            $item[$i]['index'] = $i;
            $picture = DB::table('titem_dtl')->where('id_item', $u->id_item)->select('picture')->orderBy('id_itemdtl', 'asc')->first();
            if ($picture) {
                $item[$i]['picture'] = $picture->picture;
            } else {
                $item[$i]['picture'] = null;
            }
            $item[$i]['id_item'] = $u->id_item;
            $item[$i]['vname'] = $u->vname;
            $item[$i]['vname_item'] = $u->vname_item;
            $item[$i]['id_category'] = $u->id_category;
            $item[$i]['vdescription'] = $u->vdescription;
            $item[$i]['istock'] = $u->istock;
            $item[$i]['iprice'] = $u->iprice;
            // $item[$i]['iprice_after'] = $u->iprice - ($u->iprice * $u->idisc);

            $item[$i]['iflashsale'] = $u->iflashsale;
            $item[$i]['iactive'] = $u->iactive;
        }
        return view('product_more', compact('item', 'wishlist', 'category', 'nama_web'));
    }
```

Gambar 3. 50 Function Product More *(UserPageController.php)*

Selain melakukan perubahan pada flash_sale, *Developer* melakukan perubahan untuk product yang tidak mengalami flash_sale. Cara perubahan juga sama dengan flash_sale dengan penambahan variabel wishlist yang memanggil tabel wishlist dengan kondisi id_item. Variabel ini digunakan untuk melakukan wishlist. Namun yang membedakan saat menambahkan *product* di Our Product yaitu variabel count yang mengambil data dari tabel item_hdr berisikan item dengan kondisi iflashsale = 0 dan melakukan *join* dengan tabel users agar bisa

mengambil nama toko dengan *method* GET. Dalam menggunakan *method* GET maka diperlukan *looping* dengan *foreach* agar data dapat dibaca sesuai index baris seperti pada gambar 3.50. Setelah itu akan mempassing data *item*, *wishlist* tersebut ke tampilan product_more. Agar data tersebut bisa digunakan di tampilan product_more.

```
@foreach ($item as $i => $u)
<div class="col-md-3 mb-3">
  <div class="card w-100 shadow" style="width: 18rem; height: 100%">
    @if ($u['picture'] != null)
    <a href="{{url('product-info').'/'.$u['id_item']}}">
      <img src="{{asset('assets/picture').'/'.($u['picture'])}}" class="card-img-top" alt="..." style="width:100%; height:250px" />
    </a>
    @endif
    @if ($u['picture'] == null)
    <a href="{{url('product-info').'/'.$u['id_item']}}">
      <img src="{{asset('assets/picture/blank_profilepicture.png')}}" class="card-img-top" alt="..." style="width:100%; height:250px" />
    </a>
    @endif
    <div class="card-body">
      <h5 class="card-title fw-bold"><a href="{{url('product-info').'/'.$u['id_item']}}">{{$u['vname_item']}}</a></h5>
      <p class="text-muted card-text " style="font-size: 1rem">
        {{$u['vdescription']}}
      </p>
      <p class="text-muted card-text " style="font-size: 1rem">
        {{$u['vname']}}
      </p>
      <img src="{{asset('assets/img/Review Flash Sale-01.png')}}" alt="" />
```

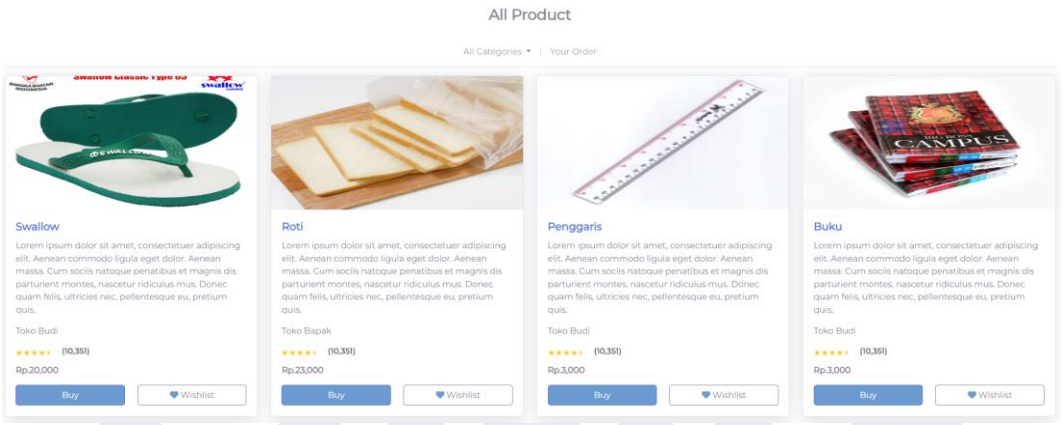
Gambar 3. 51 HTML Product_more untuk Detail product

Berikutnya di HTML product_more sama seperti HTML flash_sale perlu melakukan *foreach* dari data yang di passing dari *function* product_more seperti pada gambar 3.51. Hal ini diperlukan agar data yang bersifat *array* ini bisa ditampilkan di web browser. Selain itu, *developer* juga menambahkan *hypertext reference* untuk detail product pada bagian gambar dan nama item sehingga ketika user mengklik bagian foto atau product akan berpindah ke detail product serta menambahkan nama toko.

```
<div class="col-md text-end">
  @if (is_null($wishlist))
  <a href="/wishlist/add/{{$u['id_item']}}">
    <button type="button" class="btn btn-outline-secondary d-bloxk w-100">
      <i class="fa-solid fa-heart" style="color: #6b9bd0"></i>
      Wishlist
    </button>
  </a>
  @endif
</div>
<div class="text-muted small">
  @if (!is_null($wishlist))
  <a href="/wishlist/add/{{$u['id_item']}}">
    <button type="button" class="btn btn-outline-secondary d-bloxk w-100">
      <i class="fa-solid fa-heart" style="color: red"></i>
      Wishlist
    </button>
  </a>
  @endif
</div>
```

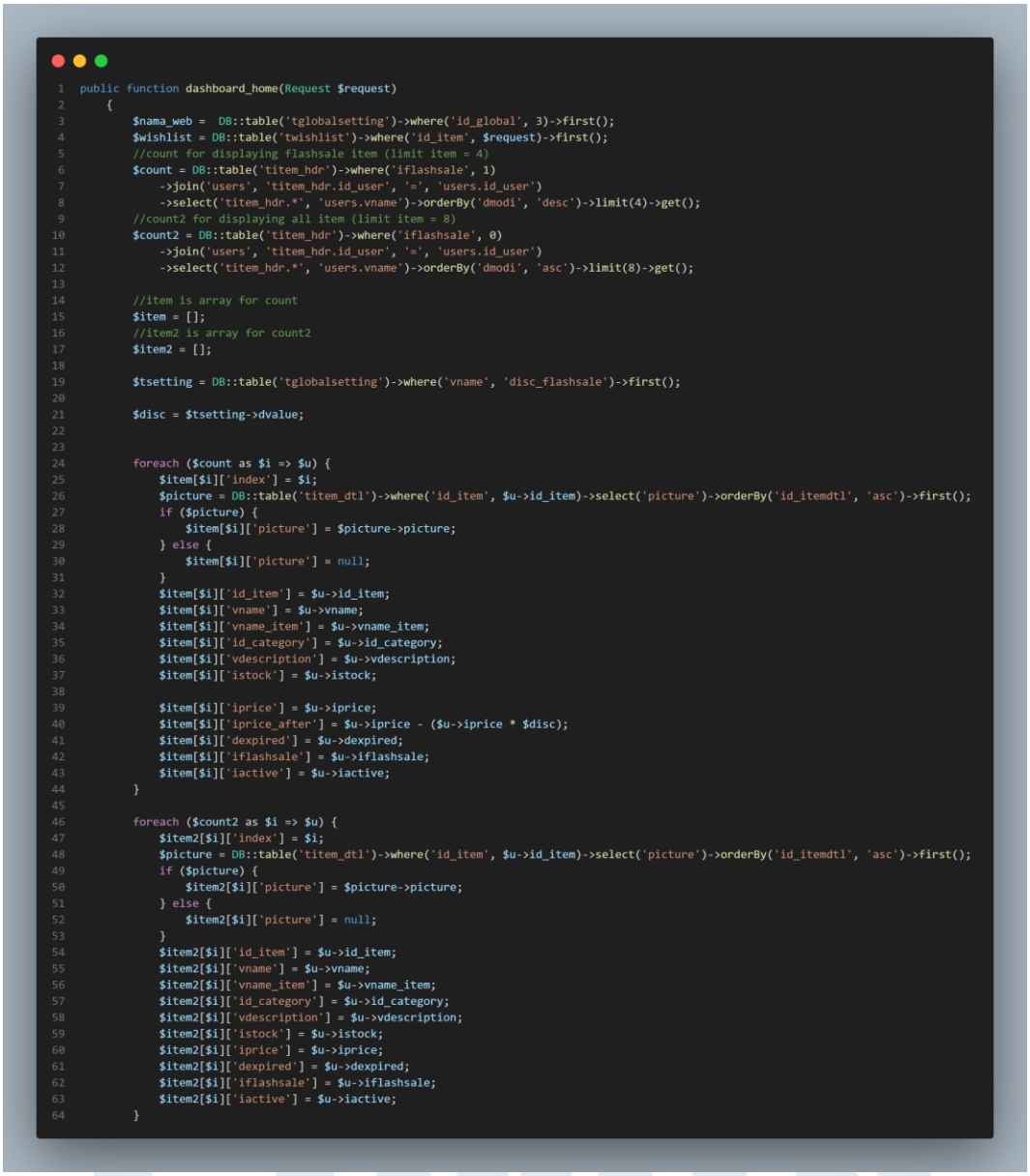
Gambar 3. 52 HTML Product More untuk wishlist

Tidak hanya itu, *developer* juga menambahkan kondisi seperti gambar 3.52 ketika tabel wishlist kosong maka wishlist tetap berwana biru tapi ketika tabel wishlist tidak kosong maka berubah warna.



Gambar 3. 53 Tampilan *Product More* yang telah diganti

Hasil dari perubahan ini dapat terlihat pada gambar 3.53. Dari hasil tersebut bahwa telah terdapat wishlist dan memudahkan user untuk menyimpan product tersebut kedalam menu wishlist.



```php
public function dashboard_home(Request $request)
    {
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        $wishlist = DB::table('twishlist')->where('id_item', $request)->first();
        //count for displaying flashsale item (limit item = 4)
        $count = DB::table('titem_hdr')->where('iflashsale', 1)
            ->join('users', 'titem_hdr.id_user', '=', 'users.id_user')
            ->select('titem_hdr.*', 'users.vname')->orderBy('dmodi', 'desc')->limit(4)->get();
        //count2 for displaying all item (limit item = 8)
        $count2 = DB::table('titem_hdr')->where('iflashsale', 0)
            ->join('users', 'titem_hdr.id_user', '=', 'users.id_user')
            ->select('titem_hdr.*', 'users.vname')->orderBy('dmodi', 'asc')->limit(8)->get();

        //item is array for count
        $item = [];
        //item2 is array for count2
        $item2 = [];

        $tsetting = DB::table('tglobalsetting')->where('vname', 'disc_flashsale')->first();

        $disc = $tsetting->dvalue;


        foreach ($count as $i => $u) {
            $item[$i]['index'] = $i;
            $picture = DB::table('titem_dtl')->where('id_item', $u->id_item)->select('picture')->orderBy('id_itemdtl', 'asc')->first();
            if ($picture) {
                $item[$i]['picture'] = $picture->picture;
            } else {
                $item[$i]['picture'] = null;
            }
            $item[$i]['id_item'] = $u->id_item;
            $item[$i]['vname'] = $u->vname;
            $item[$i]['vname_item'] = $u->vname_item;
            $item[$i]['id_category'] = $u->id_category;
            $item[$i]['vdescription'] = $u->vdescription;
            $item[$i]['istock'] = $u->istock;

            $item[$i]['iprice'] = $u->iprice;
            $item[$i]['iprice_after'] = $u->iprice - ($u->iprice * $disc);
            $item[$i]['dexpired'] = $u->dexpired;
            $item[$i]['iflashsale'] = $u->iflashsale;
            $item[$i]['iactive'] = $u->iactive;
        }

        foreach ($count2 as $i => $u) {
            $item2[$i]['index'] = $i;
            $picture = DB::table('titem_dtl')->where('id_item', $u->id_item)->select('picture')->orderBy('id_itemdtl', 'asc')->first();
            if ($picture) {
                $item2[$i]['picture'] = $picture->picture;
            } else {
                $item2[$i]['picture'] = null;
            }
            $item2[$i]['id_item'] = $u->id_item;
            $item2[$i]['vname'] = $u->vname;
            $item2[$i]['vname_item'] = $u->vname_item;
            $item2[$i]['id_category'] = $u->id_category;
            $item2[$i]['vdescription'] = $u->vdescription;
            $item2[$i]['istock'] = $u->istock;
            $item2[$i]['iprice'] = $u->iprice;
            $item2[$i]['dexpired'] = $u->dexpired;
            $item2[$i]['iflashsale'] = $u->iflashsale;
            $item2[$i]['iactive'] = $u->iactive;
        }
```

Gambar 3. 54 Function Dashboard Home (Login Controller.php)

Setelah *developer* memperbaiki tampilan flash sale dan tampilan product more, Berikutnya *developer* memperbaiki tampilan home. Perbaikan ini dilakukan dengan cara yag sama yaitu menambahkan variable wishlist yang memanggil tabel wishlist. Namun yang membedakan *function* flash_sale dan product_more dengan *function* dashboard_home ini yaitu terdapat 2 variabel yang mewakili flash_sale dan product yang tidak flash_sale. Selain itu pada tampilan home juga diberikan limit seperti flashsale diberikan limit hanya 4 product yang

akan ditampilakan di home semenatra untuk product yang tidak terkena flashsale diberi limit hanya 8 product yang ditampilkan di home dan yang lainnya akan ditampikan ditampilan product_more dan flash_sale. Pada *function* ini juga menggunakan *foreach* untuk bisa memanggil data dari tabel item_hdr dan dari function ini akan mengembalikan view home dan mempassing data dari variabel *array* item, dan wishlist dapat terlihat pada gambar 3.54.

```
<div class="card" style="width: 18rem;">
        @if ($u['picture'] != null)
        <a href="{{url('product-info').'/'.$u['id_item']}}">
          <img src="{{asset('assets/picture').'/'.($u['picture'])}}" class="card-img-top" alt="..." style="width:100%; height:250px" />
        </a>
        @endif
        @if ($u['picture'] == null)
        <a href="{{url('product-info').'/'.$u['id_item']}}">
          <img src="{{asset('assets/picture/blank_profilepicture.png')}}" class="card-img-top" alt="..." style="width:100%; height:250px" />
        </a>
        @endif
        <div class="card-body">
          <h5 class="card-title"><a href="{{url('product-info').'/'.$u['id_item']}}">{{$u['vname_item']}}</a></h5>
          <p class="text-muted card-text " style="font-size: 1rem">
            {{$u['vdescription']}}
          </p>
          <p class="text-muted card-text " style="font-size: 1rem">
            {{$u['vname']}}
          </p>
```

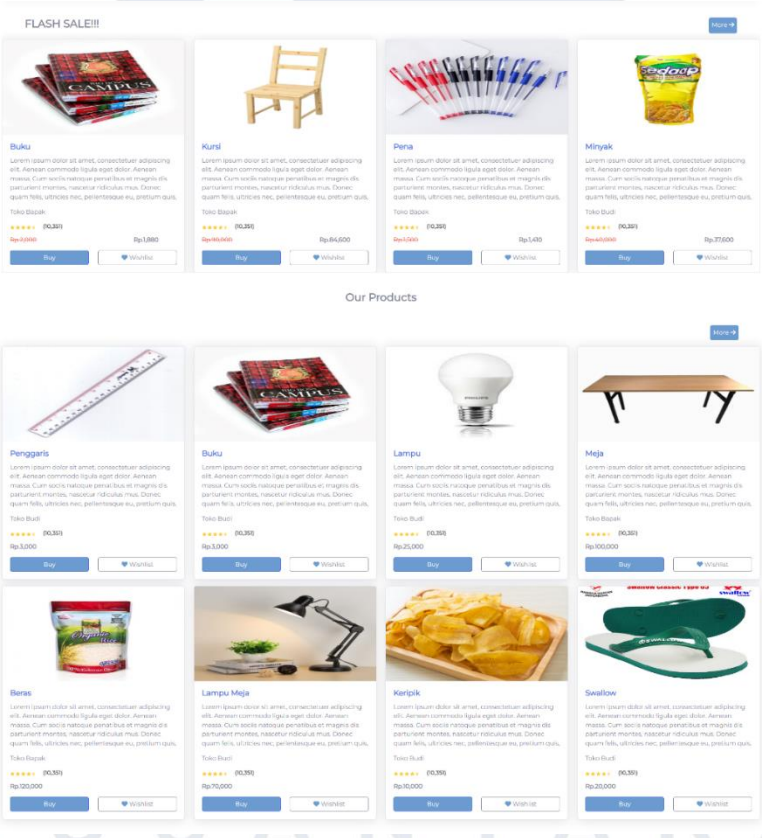
Gambar 3. 55 HTML home

Berikutnya melakukan perbaikan di tampilan HTML home seperti pada gambar 3.55. Perbaikan ini sama seperti HTML product_more dan flash_sale dengan melakukan *foreach* terlebih dahulu dari data item, lalu menambahkan URL pada bagian gambar dan nama item agar saat users mengklik gambar atau nama item akan dapat berpindah ke detail product.

```
<div class="col-md text-end">
            @if (is_null($wishlist))
            <div class="text-muted small">
              <a href="/wishlist/add/{{$u['id_item']}}">
                <button type="button" class="btn btn-outline-secondary d-bloxk w-100">
                  <i class="fa-solid fa-heart" style="color: #6b9bd0"></i>
                  Wishlist
                </button>
              </a>
            </div>
            @endif
            @if (!is_null($wishlist))
            <div class="text-muted small">
              <a href="/wishlist/add/{{$u['id_item']}}">
                <button type="button" class="btn btn-outline-secondary d-bloxk w-100">
                  <i class="fa-solid fa-heart" style="color: red"></i>
                  Wishlist
                </button>
              </a>
            </div>
            @endif
          </div>
```

Gambar 3. 56 HTML home untuk Wishlist

Tidak hanya itu saja, *developer* juga menambahkan kondisi IF untuk wishlist seperti pada gambar 3.56 seperti HTML flash_sale dan product _more.



Gambar 3. 57 Tampilan Home yang telah terubah

Hasil dari perubahan tampilan home seperti pada gambar 3.57. Dari hasil tersebut bahwa telah terdapat wihlist dan memudahkan user untuk menyimpan product tersebut kedalam menu wishlist.

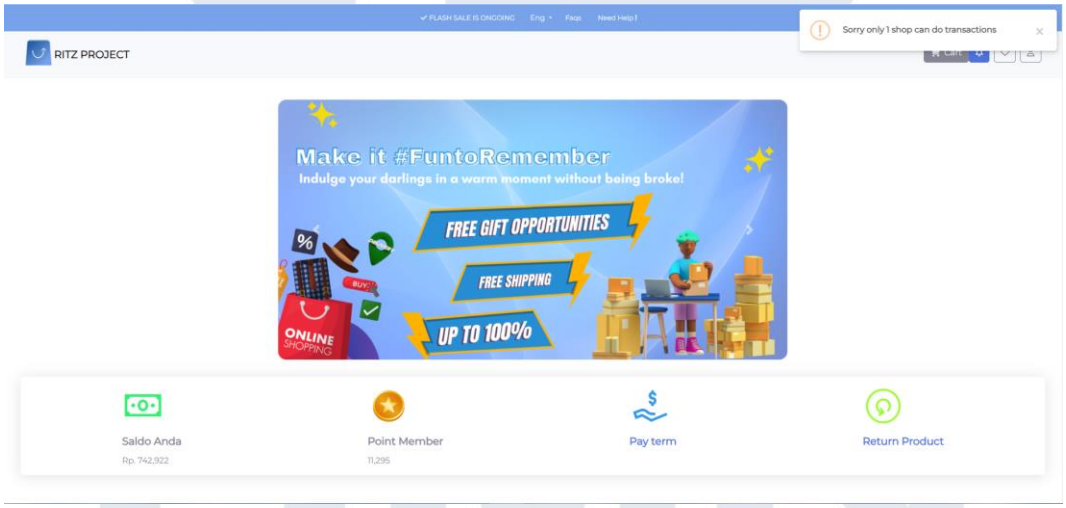
2) Pengembangan Cart Only 1 Shop

a) 4 Agustus – 9 Agustus

```php
public function add_cart($id)
{
    $check_id = DB::table('tcart')
        ->where('tcart.id_item', $id)
        ->first();

    //dd($check_id);

    $detail = DB::table('titem_hdr')
        ->join('users', 'titem_hdr.id_user', '=', 'users.id_user')
        ->where('titem_hdr.id_item', $id)
        ->first('users.id_user', Auth::user());
    //$id_store = $detail->id_user;
    //dd($id_store);
    //dd($detail);
    //dd($check_id);

    $check_id_tk = DB::table('tcart')
        ->join('titem_hdr', 'tcart.id_item', '=', 'titem_hdr.id_item')
        ->join('users', 'titem_hdr.id_user', '=', 'users.id_user')
        ->first('users.id_user', Auth::user());

    //dd($check_id_tk);
    if ($check_id == null) {
        if ($check_id_tk == null || ($detail == $check_id_tk)) {
            DB::table('tcart')->insert([
                'id_item' => $id,
                'id_user' => Auth::user()->id_user,
                'iquantity' => 1,
                'iactive' => 1,
                'vcrea' => Auth::user()->email,
                'dcrea' => Carbon::now()
            ]);
        } else if ($detail != $check_id_tk) {
            Toast('Sorry only 1 shop can do transactions', 'warning')->autoclose(3500);
            return back();
        }
    } else if ($check_id != null) {
        DB::table('tcart')->increment(
            'iquantity'
        );
    };

    return redirect('cart');
}
```

Gambar 3. 58 Function add_cart

Pada kegiatan ini *developer* diminta untuk mengubah cart yang dulunya bisa berbagai toko dari batch sebelumnya menjadi 1 toko saja. *Developer* memperbaiki dari dari function add_cart seperti pada gambar 3.58 yang memiliki *function* untuk memasukan item ke dalam keranjang atau cart. Pertama, *developer* memperbaiki variabel detail yang memanggil dari tabel item_hdr dan melakukan *join* dengan tabel user agar bisa mengambil id_user per item dan menggunakan kondisi where untuk id_item sesuai parameter request user. Lalu check_id ini memiliki fungsi untuk memanggil tabel cart yang id_item adalah *request* pengguna. *Developer* juga menambahkan variabel check_id_tk yang memanggil tabel cart dan melakukan join dengan tabel item dan users. Variabel ini berguna untuk memeriksa id_user dari item yang ingin dimasukan kedalam tabel cart.



Gambar 3. 59 Tampilan Toast di Home

Setelah itu, *developer* membuat sebuah 2 kondisi IF di dalam IF jika variabel check_id = null, lalu memeriksa variabel check_id_tk = null atau variabel detail = check_id_tk jika sesuai maka item tersebut akan dimasukan ke dalam table cart dengan perintah DML (*Data Manipulation Language*) yaitu *insert*. Jika tidak sesuai maka akan muncul notifikasi berupa toast dapat dilihat pada gambar 3.59. Lalu jika

table cart telah terisi dengan id_item yang sama akan melakukan increment quantity dari item tersebut.

3) Pengembangan Transaction for Discount Event

a) 10 Agustus – 19 Agustus

```php
public function cart()
   {
       $nama_web =  DB::table('tglobalsetting')->where('id_global',3)->first();
       $count = DB::table('tcart')
               ->where('tcart.id_user', Auth::user()->id_user)
               ->join('titem_hdr', 'tcart.id_item', '=' , 'titem_hdr.id_item')
               ->join('users', 'titem_hdr.id_user', '=', 'users.id_user')
               ->select('tcart.*', 'titem_hdr.*','users.vname')->get();
       $item =[];

       //discount for flashsale
       $tsetting_flashsale = DB::table('tglobalsetting')->where('vname','disc_flashsale')->first();
       $disc_flashsale = $tsetting_flashsale->dvalue;

       $discevent = DB::table('ttransaction_event')
           ->join('tdiscount','ttransaction_event.id_discount', '=' , 'tdiscount.id_discount' )
           ->join('tevent', 'ttransaction_event.id_event', '=', 'tevent.id_event')
           ->join('titem_hdr', 'ttransaction_event.id_item', '=', 'titem_hdr.id_item')
           ->join('users', 'ttransaction_event.id_user', '=', 'users.id_user')
           ->select('ttransaction_event.*','tdiscount.*','tevent.*', 'titem_hdr.*', 'users.vname')
           ->where('status',1)->first();

       //dd($discevent);

       $discevent2 = DB::table('ttransaction_event')
           ->join('tdiscount','ttransaction_event.id_discount', '=' , 'tdiscount.id_discount' )
           ->join('tevent', 'ttransaction_event.id_event', '=', 'tevent.id_event')
           ->join('titem_hdr', 'ttransaction_event.id_item', '=', 'titem_hdr.id_item')
           ->join('users', 'ttransaction_event.id_user', '=', 'users.id_user')
           ->where('tevent.dsendevent', '>=', Carbon::today())
           ->select('ttransaction_event.*','tdiscount.*','tevent.*', 'titem_hdr.*', 'users.vname')
           ->get();

       $total_disc_value = $discevent->value;
       $total_disc_percentage = $discevent->percentage;
```

Gambar 3. 60 Function Cart (UserPageController.php)

Pada kegiatan hari ini, *developer* didapatkan tugas untuk mengerjakan Transaction pada bagian Event. Tugas ini dimaksudkan untuk jika toko terdapat event maka event tersebut bisa digunakan oleh users. *Developer* melakukan pengembangan tugas ini di bagian *function* cart karena *function* cart ini terhubung dengan transaksi users ketika users ingin melakukan *checkout* (pembayaran).

Pertama-tama, *developer* membuat 2 variabel yaitu variabel discevent dan discevent2. Pada variabel discevent ini memanggil data dari table transaction event karena ini berisikan data event mulai dari discount, dan event. Lalu melakukan *join* dengan beberapa tabel seperti table discount, table event, table item, table users, dan menggunakan kondisi where untuk memiliki status bernilai 1 dan menggunakan *method* first. Sementara untuk variabel discevent2 ini sama seperti variabel discevent hanya dibedakan pada kondisi where yang dimana tanggal event selesai maka event tersebut tidak dapat muncul dan discevent2 menggunakan *method* GET. Setelah membuat sebuah 2 variabel, *developer* juga membuat sebuah variabel total_disc_value dan total_disc_percentage yang mengisi value dari discevent jika terisi value pada kolom value maka akan terisi ke variabel total_disc_value tapi discevent yang memiliki value pada kolom percentage maka akan terisi ke variabel total_disc_percentage seperti pada gambar 3.60.

```php
foreach ($count as $i => $u){
            $item[$i]['index'] = $i;

            $vcategory = DB::table('tcategory')->where('id_category',$u->id_category)->first();
            $picture = DB::table('titem_dtl')->where('id_item', $u->id_item)->select('picture')->orderBy('id_itemdtl', 'asc')->first();

            $price = DB::table('ttransaction_event')
            ->join('titem_hdr', 'ttransaction_event.id_item', '=', 'titem_hdr.id_item')
            ->where('ttransaction_event.id_item',333338)->first();
            //dd($price);

            if ($picture){
                $item[$i]['picture'] = $picture->picture;
            } else {
                $item[$i]['picture'] = null;
            }
            //dd($discount);

            //dd($count);
            $item[$i]['id'] = $u->id_item;
            $item[$i]['vname_item'] = $u->vname_item;
            $item[$i]['vname'] = $u->vname;
            $item[$i]['vcategory'] = $vcategory->vcategory;
            $item[$i]['iquantity'] = $u->iquantity;
            $item[$i]['iprice_after'] = $u->iprice - ($u->iprice * $disc_flashsale);
            $item[$i]['iprice_eventpercentage'] = $u->iprice - ($price->iprice * $total_disc_percentage/100);
            $item[$i]['iprice_eventvalue'] = $u->iprice - ($price->iprice - $total_disc_value);
            $item[$i]['iprice'] = $u->iprice ;
            $item[$i]['iactive'] = $u->iactive;
            //dd($item);
        }
        // $title = 'Check out!';
        // $text = "Are you sure you want to check out this item?";
        // confirmDelete($title, $text);

        return view('cart', compact('item','address','address2','member', 'min_purchase','shipping_fee','disc_birth',
        'discevent','discevent2','disc_member', 'disc_flashsale','min_point','nama_web'));
    }
```

Gambar 3. 61 Function Cart I (UserPageController.php)

Sama seperti *function* lainnya, diperlukan *method foreach* untuk bisa memanggil data dari variabel count yang memanggil data dari table cart dan melakukan join ke beberapa table seperti item, users dengan menggunakan GET. Oleh karena itu diperlukan *foreach*. Hasil dari *foreach* ini berupa *array*. *Developer* menambahkan variabel price yang memanggil dari table transaction event yang dijoinkan dengan table item agar item yang terkena diskon bisa muncul tapi masih dilakukan secara *hardcode*. Selain itu dalam *array* ini berisikan value seperti nama item, nama toko, category, quantity, harga diskon dan *developer* menambahkan 2 baris array yaitu,

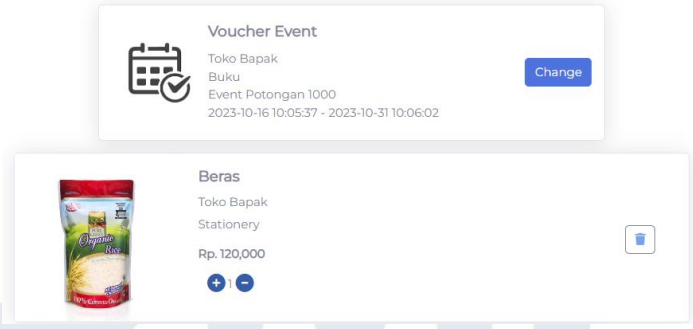
- $item[$i]['iprice_eventpercentage'] = $u->iprice - ($price->iprice * $total_disc_percentage/100);
- $item[$i]['iprice_eventvalue'] = $u->iprice - ($price->iprice - $total_disc_value);

Kedua baris ini digunakan untuk bisa menyimpan value diskon berupa angka dan percentage seperti pada gambar 3.61. Setelah itu akan mengembalikan value tersebut ke HTML cart dengan mempasssing beberapa value seperti item, discevent, discevent2 dan value lainnya.

```
<div class="col-md-8 align-items-center">
                    <h5 class="card-title fw-bold">Voucher Event</h5>
                    <p class="text-muted card-text">
                        <input
                            name="discevent"
                            type="text"
                            class="form-control"
                            value="{{$discevent->vdesc}}"
                            aria-label="Sizing example input"
                            aria-describedby="inputGroup-sizing-default"
                            hidden
                        />
                        {{$discevent->vname}}
                        <br>
                        {{$discevent->vdesc}}
                        <br>{{$discevent->dstartevent}} - {{$discevent->dsendevent}}
                    </p>
                </div>
```

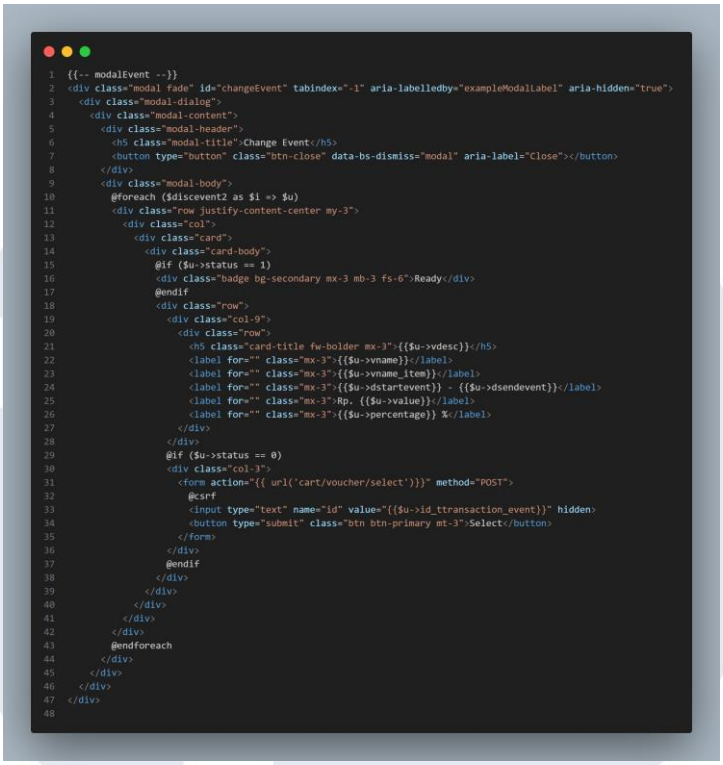
Gambar 3. 62 HTML Cart (Change Event)

Pada bagian HTML Cart, *developer* melakukan penambahan untuk bisa menampilkan voucher dan dapat memilih voucher seperti pada gambar 3.62. Dalam melakukan penambahan tersebut *developer* menggunakan value dari variabel discevent.



Gambar 3. 63 Tampilan untuk change Event

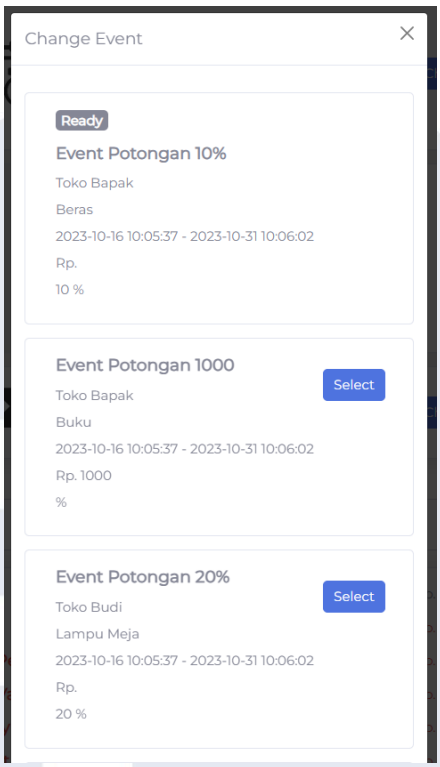
Variabel tersebut terisi nama toko, nama item, nama event, dan tanggal event dimulai dan berakhir dapat terlihat pada gambar 3.63. Selain itu untuk menggantikan event, *developer* menggunakan modal ketika user mengklik change.



```
{{-- modalEvent --}}
<div class="modal fade" id="changeEvent" tabindex="-1" aria-labelledby="exampleModalLabel" aria-hidden="true">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <h5 class="modal-title">Change Event</h5>
        <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
      </div>
      <div class="modal-body">
        @foreach ($discevent2 as $i => $u)
        <div class="row justify-content-center my-3">
          <div class="col">
            <div class="card">
              <div class="card-body">
                @if ($u->status == 1)
                <div class="badge bg-secondary mx-3 mb-3 fs-6">Ready</div>
                @endif
                <div class="row">
                  <div class="col-9">
                    <div class="row">
                      <h5 class="card-title fw-bolder mx-3">{{$u->vdesc}}</h5>
                      <label for="" class="mx-3">{{$u->vname}}</label>
                      <label for="" class="mx-3">{{$u->vname_item}}</label>
                      <label for="" class="mx-3">{{$u->dstartevent}} - {{$u->dsendevent}}</label>
                      <label for="" class="mx-3">Rp. {{$u->value}}</label>
                      <label for="" class="mx-3">{{$u->percentage}} %</label>
                    </div>
                  </div>
                  @if ($u->status == 0)
                  <div class="col-3">
                    <form action="{{ url('cart/voucher/select')}}" method="POST">
                      @csrf
                      <input type="text" name="id" value="{{$u->id_ttransaction_event}}" hidden>
                      <button type="submit" class="btn btn-primary mt-3">Select</button>
                    </form>
                  </div>
                  @endif
                </div>
              </div>
            </div>
          </div>
        </div>
        @endforeach
      </div>
    </div>
  </div>
</div>

```

Gambar 3. 64 Modal Change Event

Gambar 3.64 merupakan modal untuk menggantikan event. *Developer* menggunakan variabel discevent2 yang menggunakan GET maka dari itu diperlukan *foreach* untuk bisa menggunakan variabel discevent2. Jika status event bernilai 1 maka terdapat tulisan ready. Jika bernilai 0 maka terdapat tulisan select dan terhubung dengan URL *route* dengan *method* POST.



Gambar 3. 65 Modal untuk menggantikan Event

Selain itu pada modal ini juga menampilkan nama toko, nama item, nama event, value dari kolom percentage, value dari kolom value, tanggal mulai event dan tanggal berakhirnya dapat terlihat pada gambar 3.65.

```
1  Route::post('cart/voucher/select','App\Http\Controllers\UserPageController@cart_voucher_select');
2
```

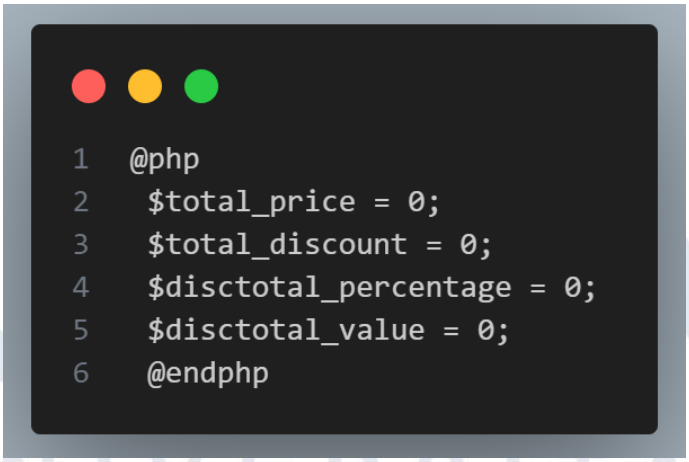
Gambar 3. 66 Route untuk Select Event

Gambar 3.66 merupakan route dari sebuah function untuk memilih event dengan adanya route ini membuat function dan request user terhubung dengan baik.

```php
public function cart_voucher_select(Request $request)
    {
        DB::table('ttransaction_event')->where('status', 1)
            ->update([
                'status' => 0
            ]);

        DB::table('ttransaction_event')->where('id_ttransaction_event', $request->id)
            ->update([
                'status' => 1
            ]);

        return back();
    }
```

Gambar 3. 67 Function Change Event

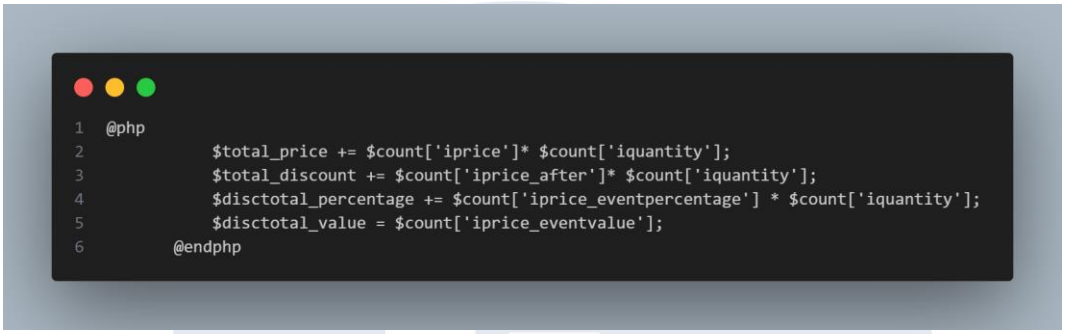
Agar ini bisa berjalan dengan baik tentu harus memiliki *function* dan *route*, maka dari itu developer membuat sebuah function untuk select event sesuai *request*. Ketika status dari event terserbut bernilai 1 maka akan diupdate menjadi bernilai 0 dan ketika users memilih event maka status event yang bernilai 0 akan diupdate menjadi 1 seperti gambar 3.67.



```php
@php
 $total_price = 0;
 $total_discount = 0;
 $disctotal_percentage = 0;
 $disctotal_value = 0;
 @endphp
```

Gambar 3. 68 HTML Cart I

Setelah developer membuat untuk pilihan event, Berikutnya *developer* membuat inisialisasi 2 variabel yaitu disctotal_percentage dan disc_value menggunakan php pada HTML seperti pada gambar 3.68 untuk bisa menampung nilai discount event.



```
@php
            $total_price += $count['iprice']* $count['iquantity'];
            $total_discount += $count['iprice_after']* $count['iquantity'];
            $disctotal_percentage += $count['iprice_eventpercentage'] * $count['iquantity'];
            $disctotal_value = $count['iprice_eventvalue'];
        @endphp
```

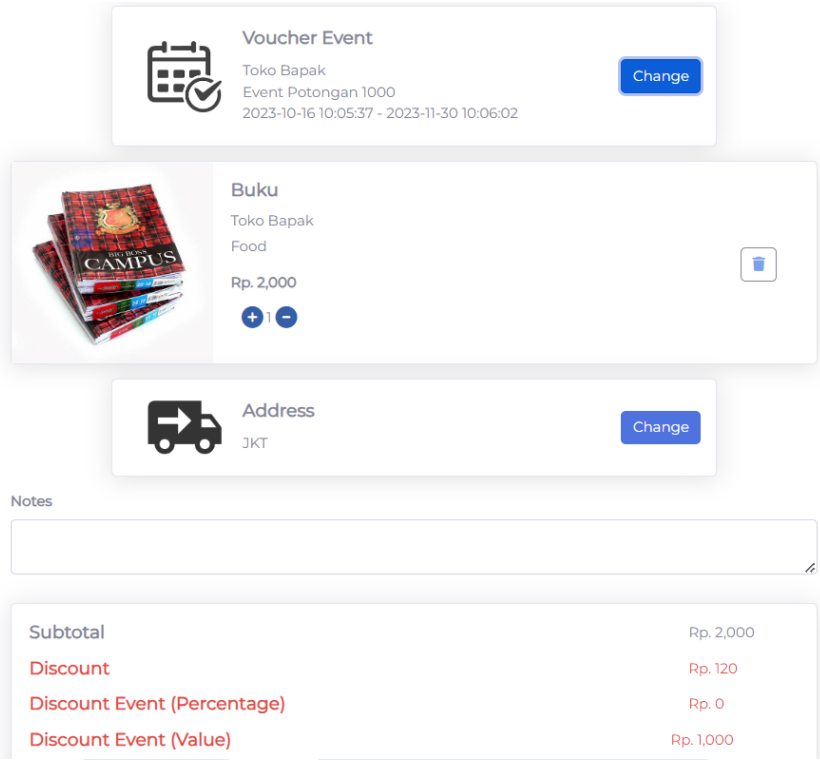
Gambar 3. 69 HTML Cart II

Lalu setelah melakukan *foreach* item maka perlu juga menggunakan PHP untuk mengisi value tersebut kedalam varaibel yang telah diinisialisasi. Dengan begitu variabel tersebut bisa digunakan untuk menampilkan nilai value diskon seprti pada gambar 3.69.

```
<div class="row">
                        <div class="col-md-10">
                          <h5 class="card-text fw-bold text-danger">Discount Event (Percentage)</h5>
                        </div>
                        <div class="col-md-2">
                          <p class="text-danger">Rp. {{number_format($total_price - $disctotal_percentage)}}</p>
                        </div>
                        <div class="row">
                        <div class="col-md-10">
                          <h5 class="card-text fw-bold text-danger">Discount Event (Value)</h5>
                        </div>
                        <div class="col-md-2">
                          <p class="text-danger">Rp. {{number_format($disctotal_value)}}</p>
                        </div>
```

Gambar 3. 70 HTML Cart Discount Event

Gambar 3.70 merupakan cara agar bisa menampilkan nilai *discount* seperti untuk nilai *discount* percentage maka nilai dari total harga akan dikurangi dari variabel yang telah dibuat yaitu disctotal_percentage.



Gambar 3. 71 Tampilan Discount Event

Sementara jika *discount* berupa value atau harga maka akan terisi value di bagian *discount* event value sesuai potongan valuenya dapat dilihat pada gambar 3.71.

4) Pengembangan Transaction for Discount Birthday

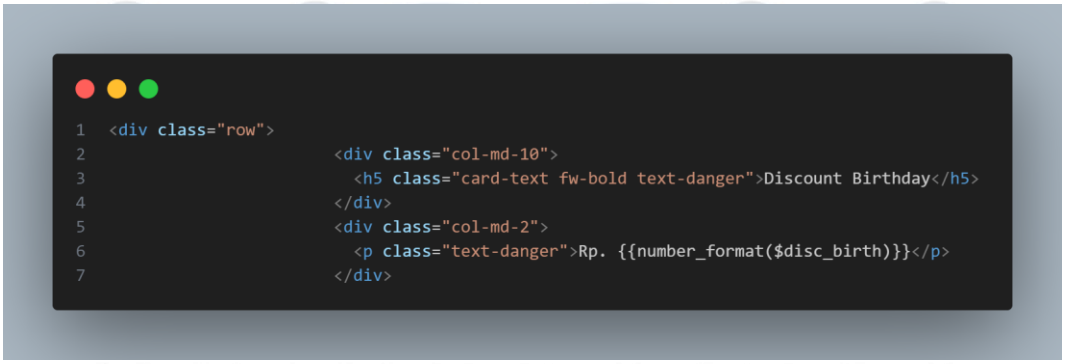
a) 21 Agustus – 26 Agustus

```php
$today = now();
        $birthday = DB::table('users')
            ->where('id_user', Auth::user()->id_user)
            ->whereMonth('dbirthday', $today->month)
            ->whereDay('dbirthday', $today->day)
            ->select('users.dbirthday')->first();
        //dd($birthday);
        if ($birthday != null) {
            $tsetting_birth = DB::table('tglobalsetting')->where('vname', 'disc_birthday')->first();
            $disc_birth = $tsetting_birth->ivalue;
            //dd($disc_birthday);
        } else {
            $disc_birth = 0;
        }
```

Gambar 3. 72 Code untuk membacakan Ulang Tahun

Pada kegitan ini *developer* ditugaskan untuk membuat diskon bagi user yang berulang tahun bahwa mendapatkan potongan sebesar Rp. 10.000. Oleh karena itu, developer menggunakan *Method* Carbon now untuk bisa mengambil tanggal sekarang in kedalam *variable* today. Lalu *developer* membuat variabel dengan nama birthday yang mengambil data dari table users, lalu menggunakan kondisi where id_user sesuai dengan id_user yang login dan menggunakan 2 kondisi yaitu whereMonth agar bisa mengambil bulan dari tanggal lahir user yang disusaikan dengan bulan dari tanggal tersebut. Tidak hanya menggunakan WhereMonth saja tapi juga menggunakan WhereDays untuk menggambil tanggal lahir user.

Berikutnya membuat sebuah validation jika variabel birthday tidak null maka akan mengambil data dari tabel globalsetting yang berisikan value diskon ulang tahun ke dalam variabel tsetting_birth dan membuat sebuah satu variabel disc_birth agar dapat menyimpan value discount birthday tapi jika birthday null kak variabel disc_birth akan bernilai 0 seperti pada gambar 3.72. Nilai dari disc_birth ini akan dipassing ke tampilan cart



Gambar 3. 73 HTML untuk Disocunt Birthday pada bagian Cart

Pada bagian HTML cart, *developer* menambahakna 1 baris untuk bisa menampilkan value discount_birthday. *Developer* tinggal memasukan nilai variabel disc_birth yang telah dibuat di *function*

Gambar 3. 74 Tampilan ketika user tidak birthday

Maka hasil tampilan akan seperti pada gambar 3.74 bagi *user* yang sedang tidak berulang tahun

Gambar 3. 75 Tampilan ketika user birthday

Pada gambar 3.75 merupakan tampilan bagi *user* yang sedang berulang tahun akan mendapatkan potongan diskon sebesar 10.000.

5) Pengembangan Transaction for Discount Member

a) 28 Agustus – 2 September

```
$member = DB::table('tmember')->where('id_user', Auth::user()->id_user)->first();
        $tsetting_member = DB::table('tglobalsetting')->where('vname', 'disc_member')->first();
        $disc_member = $tsetting_member->ivalue;
        $tsetting_minpurchase = DB::table('tglobalsetting')->where('vname', 'min_purchase')->first();
        $min_purchase = $tsetting_minpurchase->ivalue;

```

Gambar 3. 76 Code discount Member

Pada kegiatan ini, *developer* juga mendapatkan tugas untuk membuat *discount* member dengan ada batasan minimal transaksi dan user telah menjadi member. Pertama-tama, *developer* membuat sebuah variabel dengan nama member untuk bisa mendapatkan data dari tabel

member dengan kondisi where id_user adalah id yang sedang login di *web browser e-commerce*. Lalu *developer* juga membuat sebuah variabel tsetting_member untuk bisa memanggil data dari table global setting dengan kondisi where kolom vname dengan nama baris disc_member. Setelah mendapatkan data maka akan mengisi value tersebut kedalam variabel disc_member seperti $disc_member = $tsetting_member->ivalue.

Tidak hanya disc_member saja tapi juga perlu mengambil data minimal pembelian. Dengan cara yang hampir sama dengan mengambil data *discount* member. Variabel tsetting_minpurchase dari tabel global setting dengan kondisi where kolom vname dengan nama baris min_purchase dapat terlihat pada gambar 3.76. Lalu data tersebut dimasukan kedalam variabel min_purchase seperti $min_purchase = $tsetting_minpurchase->ivalue. Variabel disc_member dan min_purchase ini yang dipassing ke HTML cart.

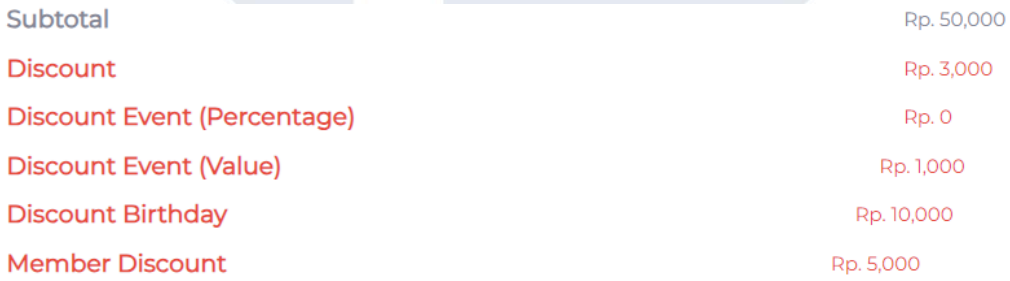
```
<div class="row">
                    <div class="col-md-10">
                      <h5 class="card-text fw-bold text-danger">
                        Member Discount
                      </h5>
                    </div>
                    <div class="col-md-2">
                      @if (is_null($member))
                        <p class="text-danger">Rp. 0</p>
                      @endif

                      @if (!is_null($member))
                        @if($total_price >= $min_purchase)
                          @if ($member->istatus_member == 1)
                            <p class="text-danger">Rp. {{number_format($disc_member)}}</p>
                          @endif
                          @if ($member->istatus_member == 0)
                            <p class="text-danger">Rp. 0</p>
                          @endif
                        @endif

                        @if($total_price <= $min_purchase)
                          <p class="text-danger">Rp. 0</p>
                        @endif
                      @endif
                    </div>
                  </div>
```

Gambar 3. 77 HTML Discount member di Cart

Pada HTML maka terdapat baris baru dengan nama member *discount* seperti pada gambar 3.77. *Developer* membuat beberapa validation dengan kondisi IF jika variabel member bernilai null maka pada *discount* member bernilai 0.

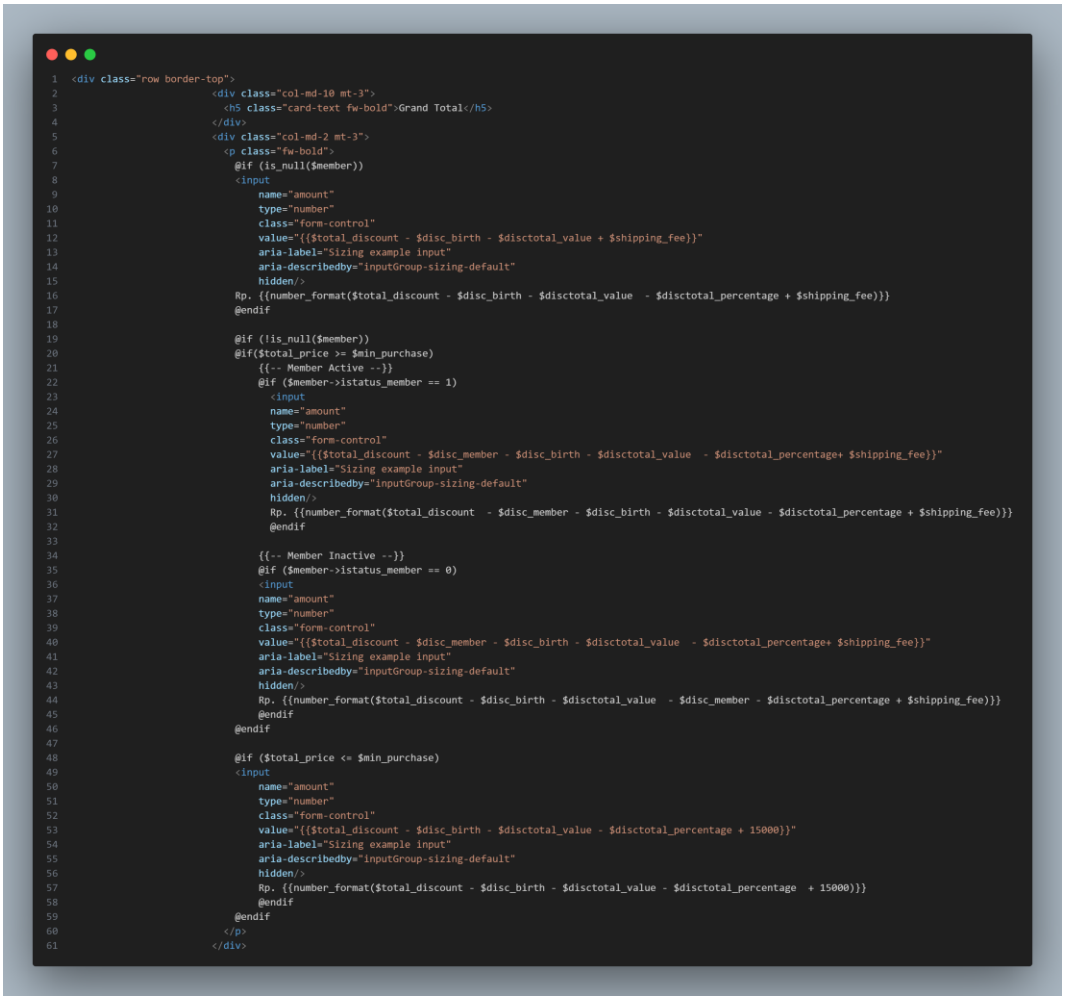
Gambar 3. 78 Tampilan user tidak mendapatkan Discount member

Seperti pada gambar 3.78 tapi jika bernilai 0 maka tidak mendapatkan member discount begitu juga jika total pembelian tidak memenuhi minimal pembelian juga akan bernilai nol.



Gambar 3. 79 Tampilan user mendaptkan Discount member

Namun, jika member tidak null maka akan melihat kondisi apakah total price (total pembelian) telah lebih besar dari minimal pembelian jika telah sesuai maka akan melihat status member tersebut jika bernilai 1 maka baru mendapatkan discount member seperti pada gambar 3.79.



```
<div class="row border-top">
                        <div class="col-md-10 mt-3">
                          <h5 class="card-text fw-bold">Grand Total</h5>
                        </div>
                        <div class="col-md-2 mt-3">
                          <p class="fw-bold">
                            @if (is_null($member))
                            <input
                                name="amount"
                                type="number"
                                class="form-control"
                                value="{{$total_discount - $disc_birth - $disctotal_value + $shipping_fee}}"
                                aria-label="Sizing example input"
                                aria-describedby="inputGroup-sizing-default"
                                hidden/>
                            Rp. {{number_format($total_discount - $disc_birth - $disctotal_value  - $disctotal_percentage + $shipping_fee)}}
                            @endif

                            @if (!is_null($member))
                            @if($total_price >= $min_purchase)
                                {{-- Member Active --}}
                                @if ($member->istatus_member == 1)
                                  <input
                                  name="amount"
                                  type="number"
                                  class="form-control"
                                  value="{{$total_discount - $disc_member - $disc_birth - $disctotal_value  - $disctotal_percentage+ $shipping_fee}}"
                                  aria-label="Sizing example input"
                                  aria-describedby="inputGroup-sizing-default"
                                  hidden/>
                                  Rp. {{number_format($total_discount  - $disc_member - $disc_birth - $disctotal_value - $disctotal_percentage + $shipping_fee)}}
                                  @endif

                                {{-- Member Inactive --}}
                                @if ($member->istatus_member == 0)
                                <input
                                name="amount"
                                type="number"
                                class="form-control"
                                value="{{$total_discount - $disc_member - $disc_birth - $disctotal_value  - $disctotal_percentage+ $shipping_fee}}"
                                aria-label="Sizing example input"
                                aria-describedby="inputGroup-sizing-default"
                                hidden/>
                                Rp. {{number_format($total_discount - $disc_birth - $disctotal_value  - $disc_member - $disctotal_percentage + $shipping_fee)}}
                                @endif
                            @endif

                            @if ($total_price <= $min_purchase)
                            <input
                                name="amount"
                                type="number"
                                class="form-control"
                                value="{{$total_discount - $disc_birth - $disctotal_value - $disctotal_percentage + 15000}}"
                                aria-label="Sizing example input"
                                aria-describedby="inputGroup-sizing-default"
                                hidden/>
                                Rp. {{number_format($total_discount - $disc_birth - $disctotal_value - $disctotal_percentage  + 15000)}}
                                @endif
                            @endif
                          </p>
                        </div>
```

Gambar 3. 80 HTML Transaction Grand Total di Cart

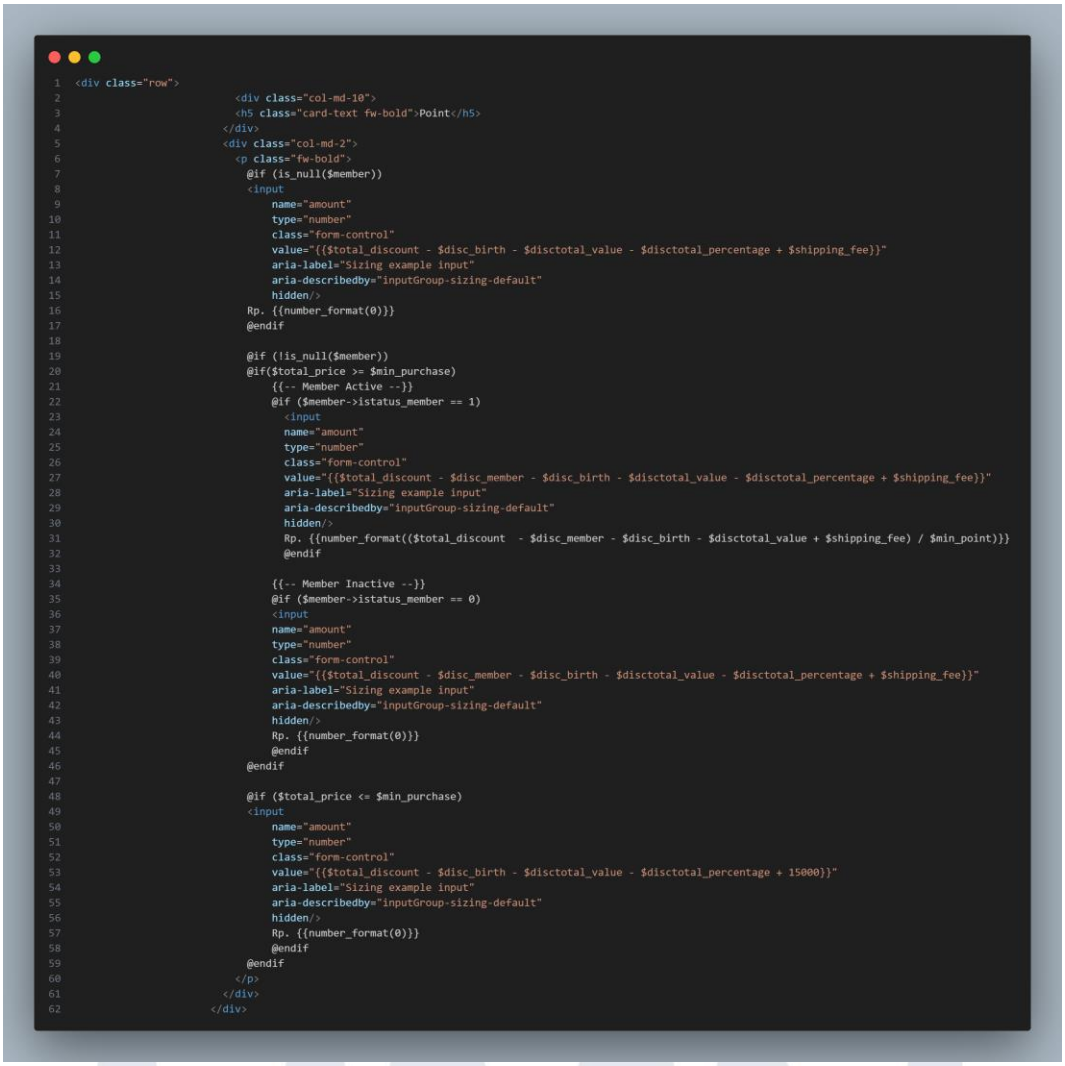
Dikarenakan adanya pengembangan pada member dan minimal pembelian maka pada grand total juga mengalami perbaikan karena dipengaruhi dari member. *Developer* melakukan perbaikan dengan menambahkan kondisi *validation* IF. Jika variabel member bernilai *null* maka grand total tidak akan mengurangi *discount* member tapi jika member tidak null dan *total price* (total pembelian) telah mencapai minimal pembelian maka jika member tersebut masih *active* maka grand total akan mengurangi discount member tapi jika member tidak *active* maka grand total tidak akan mengurangi *discount* member. Begitu juga jika total pembelian tidak sesuai dengan minimal pembelian maka grand total tidak akan dikurangi dengan *discount* member.

6) Pengembangan Transaction for Point Member

a) 4 September – 9 September

Gambar 3. 81 Code Point Member

Pada kegiatan ini, *developer* ditugaskan untuk membuat sebuah point member setelah berhasil mengembangkan *transaction discount* member. Pertama-tama, *developer* membuat sebuah variabel tsetting_point untuk mengambil data dari table global setting dengan kondisi where dari kolom vname dan baris calc_point. Lalu terdapat variabel min_point yang berisikan value dari tsetting_point. Selain itu, dibutuhkan juga *variable* member yang telah dibuat sebelumnya. Variabel min_point ini yang akan dipassing ke HTML cart.



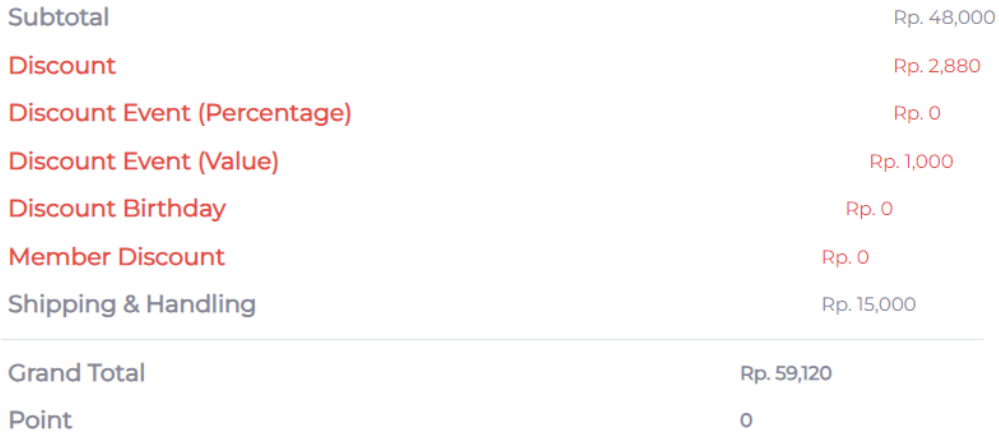
```
<div class="row">
                        <div class="col-md-10">
                        <h5 class="card-text fw-bold">Point</h5>
                      </div>
                      <div class="col-md-2">
                        <p class="fw-bold">
                          @if (is_null($member))
                          <input
                              name="amount"
                              type="number"
                              class="form-control"
                              value="{{$total_discount - $disc_birth - $disctotal_value - $disctotal_percentage + $shipping_fee}}"
                              aria-label="Sizing example input"
                              aria-describedby="inputGroup-sizing-default"
                              hidden/>
                          Rp. {{number_format(0)}}
                          @endif

                          @if (!is_null($member))
                          @if($total_price >= $min_purchase)
                              {{-- Member Active --}}
                              @if ($member->istatus_member == 1)
                                <input
                                name="amount"
                                type="number"
                                class="form-control"
                                value="{{$total_discount - $disc_member - $disc_birth - $disctotal_value - $disctotal_percentage + $shipping_fee}}"
                                aria-label="Sizing example input"
                                aria-describedby="inputGroup-sizing-default"
                                hidden/>
                                Rp. {{number_format(($total_discount  - $disc_member - $disc_birth - $disctotal_value + $shipping_fee) / $min_point)}}
                                @endif

                              {{-- Member Inactive --}}
                              @if ($member->istatus_member == 0)
                              <input
                              name="amount"
                              type="number"
                              class="form-control"
                              value="{{$total_discount - $disc_member - $disc_birth - $disctotal_value - $disctotal_percentage + $shipping_fee}}"
                              aria-label="Sizing example input"
                              aria-describedby="inputGroup-sizing-default"
                              hidden/>
                              Rp. {{number_format(0)}}
                              @endif
                          @endif

                          @if ($total_price <= $min_purchase)
                          <input
                              name="amount"
                              type="number"
                              class="form-control"
                              value="{{$total_discount - $disc_birth - $disctotal_value - $disctotal_percentage + 15000}}"
                              aria-label="Sizing example input"
                              aria-describedby="inputGroup-sizing-default"
                              hidden/>
                              Rp. {{number_format(0)}}
                              @endif
                          @endif
                        </p>
                      </div>
                    </div>
```

Gambar 3. 82 HTML Point di Cart

Berikutnya *developer* menambahkan baris untuk point pada tampilan Cart. *Developer* melakukan *validation* dengan kondisi IF dapat terlihat pada gambar 3.82. Jika variabel member bernilai *null* maka point yang didapatkan bernilai 0

Gambar 3. 83 Tampilan user mendapatkan point di bagian Cart

Namun, jika member tidak *null* dan *total price* (total pembelian) telah mencapai minimal pembelian dan jika member tersebut masih *active* maka akan mendapatkan point dari hasil grand total dibagi dari min_point seperti gambar 3.83



Gambar 3. 84 Tampilan user tidak mendapatkan point di bagian Cart

Namun, jika member tidak active maka point yang didapatkan bernilai 0. Begitu juga jika total pembelian tidak sesuai dengan minimal pembelian maka point akan bernilai 0 seperti gambar 3.84.

7) Pengembangan Transaction for Payterm

a) 11 Sepetmber – 16 September

```php
class XenditController extends Controller
{
    public function store(Request $request)
    {

        $secret_key = 'Basic ' . config('xendit.key_auth');
        $external_id = Str::random(10);

        $data_request = Http::withHeaders([
            'Authorization' => $secret_key
        ])->post('https://api.xendit.co/v2/invoices', [
            'external_id' => $external_id,
            'amount' => $request->amount
        ]);

        $response = $data_request->object();
        $cart_data = DB::table('tcart')->where('tcart.id_user', Auth::user()->id_user)
            ->join('titem_hdr', 'tcart.id_item', '=', 'titem_hdr.id_item')
            ->join('users', 'titem_hdr.id_user', '=', 'users.id_user')->get();
        $count_data = DB::table('tcart')->where('id_user', Auth::user()->id_user)->count();
        $item = [];
        $total_price = 0;

        foreach ($cart_data as $i => $u) {
            $item[$i]['iquantity'] = $u->iquantity;
            $item[$i]['iprice'] = $u->iprice;
        }
        foreach ($item as $count) {
            $total_price += $count['iprice'] * $count['iquantity'];
        }

        Payment::create([
            'vsecret_code' => $external_id,
            'vdescription' => $request->notes,
            'ipayment_status' => $response->status,
            'vpayment_link' => $response->invoice_url,
            'iamount' => $request->amount,
            'vcrea' => Auth::user()->email,
            'dcrea' => Carbon::now(),
        ]);

        $check_payment = DB::table('tpayment')->where('dcrea', Carbon::now())->first();

        $member = DB::table('tmember')->where('id_user', Auth::user()->id_user)->first();

        $tsetting_minpurchase = DB::table('tglobalsetting')->where('vname', 'min_purchase')->first();
        $min_purchase = $tsetting_minpurchase->ivalue;
```

Gambar 3. 85 Code Checkout di XenditController.php

Pada kegiatan ini *developer* ditugas untuk membuat payterm ketika user telah menjadi member dan memenuhi minimal pembelian. Gambar 3.85 merupakan potongan *function* store setelah user melakukan check out. *Function* ini berada di *Xendit Controller*. Dalam *payterm* ini users bisa melakukan pembayaran *payterm* sebanyak inputan dari admin toko yang menginput dalam tabel global setting. Dalam pengerjaan tugas ini minimal dalam melakukan payterm sebanyak 10 kali.

```php
$member_term = DB::table('tmember')
            ->where('id_user', Auth::user()->id_user)
            ->where('istatus_member', 1)->first();

        $tsetting8 = DB::table('tglobalsetting')->where('vname', 'min_term')->first();
        $min_term = $tsetting8->ivalue;
        $lop = 0;

        $check_transaction = DB::table('ttransaction_hdr')->where('dcrea', Carbon::now())->first();

        if (!is_null($check_transaction)) {
            //Non Member
            if (is_null($member)) {
                foreach ($cart_data as $i) {
                    DB::table('titem_hdr')->where('id_item', $i->id_item);
                    DB::table('titem_hdr')->where('id_item', $i->id_item)->decrement('istock', $i->iquantity);
                    DB::table('ttransaction_dtl')->insert([
                        'id_transaction' => $check_transaction->id_transaction,
                        'id_item' => $i->id_item,
                        'iquantity' => $i->iquantity,
                        'iprice' => ($i->iprice * $i->iquantity),
                        'ipoint' => 0,
                        'dspack' => Carbon::now(),
                        'vcrea' => Auth::user()->email,
                        'dcrea' => Carbon::now(),
                    ]);
                }
                DB::table('tcart')->where('id_user', Auth::user()->id_user)->delete();
            }
            // Member
            if (!is_null($member)) {
                if ($total_price >= $min_purchase) {
                    if ($member->istatus_member == 1) {
                        foreach ($cart_data as $i) {
                            DB::table('titem_hdr')->where('id_item', $i->id_item)->first();
                            DB::table('titem_hdr')->where('id_item', $i->id_item)->decrement('istock', $i->iquantity);
                            DB::table('ttransaction_dtl')->insert([
                                'id_transaction' => $check_transaction->id_transaction,
                                'id_item' => $i->id_item,
                                'iquantity' => $i->iquantity,
                                'iprice' => ($i->iprice * $i->iquantity),
                                'ipoint' => ($i->iprice * $i->iquantity) / $min_point,
                                'dspack' => Carbon::now(),
                                'vcrea' => Auth::user()->email,
                                'dcrea' => Carbon::now(),
                            ]);
                        }
                        for ($lop = 1; $lop <= $min_term; $lop++) {
                            DB::table('tmember_dtl')->insert([
                                'id_member' => $member_term->id_member,
                                'id_transaction' => $check_transaction->id_transaction,
                                'iterm' => $check_transaction->itotal_price / $min_term,
                                'vcrea' => Auth::user()->email,
                                'dcrea' => Carbon::now(),
                            ]);
                        }
                        DB::table('tcart')->where('id_user', Auth::user()->id_user)->delete();
                    }
                } elseif ($member->istatus_member == 0) {
                    foreach ($cart_data as $i) {
                        DB::table('titem_hdr')->where('id_item', $i->id_item);
                        DB::table('titem_hdr')->where('id_item', $i->id_item)->decrement('istock', $i->iquantity);
                        DB::table('ttransaction_dtl')->insert([
                            'id_transaction' => $check_transaction->id_transaction,
                            'id_item' => $i->id_item,
                            'iquantity' => $i->iquantity,
                            'iprice' => ($i->iprice * $i->iquantity),
                            'ipoint' => 0,
                            'dspack' => Carbon::now(),
                            'vcrea' => Auth::user()->email,
                            'dcrea' => Carbon::now(),
                        ]);
                    }
                    DB::table('tcart')->where('id_user', Auth::user()->id_user)->delete();
                }
            }
            if ($total_price <= $min_purchase) {
                foreach ($cart_data as $i) {
                    DB::table('titem_hdr')->where('id_item', $i->id_item);
                    DB::table('titem_hdr')->where('id_item', $i->id_item)->decrement('istock', $i->iquantity);
                    DB::table('ttransaction_dtl')->insert([
                        'id_transaction' => $check_transaction->id_transaction,
                        'id_item' => $i->id_item,
                        'iquantity' => $i->iquantity,
                        'iprice' => ($i->iprice * $i->iquantity),
                        'ipoint' => 0,
                        'dspack' => Carbon::now(),
                        'vcrea' => Auth::user()->email,
                        'dcrea' => Carbon::now(),
                    ]);
                }
                DB::table('tcart')->where('id_user', Auth::user()->id_user)->delete();
            }
```

Gambar 3. 86 Code untuk payterm di Xendit Controller

Oleh karena itu *developer* melakukan *looping* dari data transaction_hdr karena dalam *payterm* mengambil data berupa grand total yang dibayarkan oeh users dibagi dengan minimal term. Pertama-tama, *Developer* menambahkan 2 join pada variabel cart_data yaitu *join* dengan table item hdr dan table user. Lalu *developer* melakukan *foreach* agar dari variabel cart_data bisa digunakan sesuai kebutuhan *developer*. Pada *foreach* ini *developer* membutuhkan *quantity* dan harga item untuk bisa mendapatkan total_price dari harga barang dikalikan dengan jumlah quantity. Berikutanya *developer* membuat sebuah variabel member untuk memanggil data member dan membuat variabel min_purchase dari data global setting. Hal ini dibutuhkan karena telah dijelaskan sebelumnya bahwa payterm user yang mendapatkan fitur *payterm* bagi user yang telah menjadi member dan telah memenuhi minimal pembelian.

Berikutnya *developer* membuat sebuah variabel dengan nama member_term yang memanggil *table* member dengan 2 kondisi where yaitu id_user adalah id_user yang login dan status member bernilai 1. Berikutnya *developer* memanggil data dari tabel global setting dengan nama kolom vname dan baris min_term kedalam *variable* tsetting8 dan dari varibel tersebut akan diambil value minimal *term* kedalam variabel min_term. *Developer* juga mempersiapkan variabel lop untuk dijadikan *looping*.

Setelah itu dibutuhkan juga *variable* yang memeriksa data table transaction_hdr apakah tidak *null*, jika tidak null maka ada beberapa kondisi IF yang harus dipenuhi agar bisa melakukan *looping* data *payterm*. Pertama jika member *null* mka tidak akan melakukan *looping payterm* hanya memasukan data transaction detail seperti id_item, quantity, harga kedalam *table* transaction_dtl. Namun, jika member tidak null dan total price telah memenuhi minimal pembelian serta status member bernilai 1 maka akan melakukan *looping payterm* dengan

method for yang berarti jika variabel lop kurang dari min_term maka akan terus melakukan *lopping* ke dalam tabel member_dtl yang berisikan data id_member, id_transaction, dan total pembayaran user yang dibagi dengan min_term. Selain itu juga melakukan *insert* data ke transaction_dtl sesuai transaksi user. Jika member memiliki status bernilai 0 maka tidak mendapatkan payterm. Begitu juga jika total price kurang dari minimal transaksi maka tidak mendapatkan fitur pay term seperti pada gambar 3.86.

```php
public function transaction_payment()
    {
        $item_detail = null;
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        $transaction = DB::table('ttransaction_hdr')->where('id_user', Auth::user()->id_user)
            ->join('tpayment', 'ttransaction_hdr.id_payment', '=', 'tpayment.id_payment')
            ->select('ttransaction_hdr.*', 'tpayment.ipayment_status', 'tpayment.vpayment_link')
            ->get();

        $member = DB::table('tmember')->where('id_user', Auth::user()->id_user)->first();
        $tsetting_member = DB::table('tglobalsetting')->where('vname', 'disc_member')->first();

        foreach ($transaction as $item => $u) {
            $item_detail = DB::table('ttransaction_dtl')
                ->join('titem_hdr', 'ttransaction_dtl.id_item', '=', 'titem_hdr.id_item')
                ->select('ttransaction_dtl.*', 'titem_hdr.vname_item')
                ->get();
        }
        if (is_null($item_detail)) {
            return view('transaction_payment', compact('nama_web', 'transaction', 'member'));
        }
        if (!is_null($item_detail)) {
            return view('transaction_payment', compact('nama_web', 'transaction', 'item_detail', 'member'));
        }
    }
```

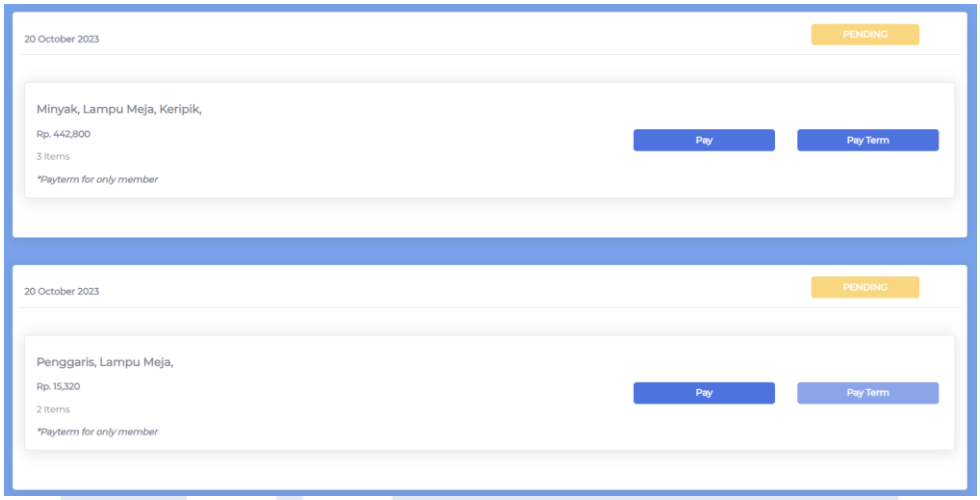
Gambar 3. 87 Function Transaction_payment

Setelah membuat transaksi untuk *payterm* maka selanjutnya membuat tampilan untuk mengambilkan data *payterm*. *Developer* membuat tampilan tersebut ada tampilan transaction payment maka dari itu *developer* melakukan penambahan code pada *function* transaction *payment*. *Developer* menambahkan variabel member untuk bisa memanggil data member karena *payterm* ini akan aktif jika user telah menjadi member. *Variable* member tersebut yang dipassing ke tampilan transaction_payment dapat terlihat pada gambar 3.87

```
<div class="col-md-2 d-flex align-items-center justify-content-center me-3">
                    @if (is_null($member))
                        <a href= '#' style="width: inherit;">
                            <button class="btn btn-primary w-100 fw-bold" disabled >Pay Term</button>
                        </a>
                    @endif

                    @if (!is_null($member))
                      {{-- Member Active --}}
                        @if ($member->istatus_member == 1)
                          <a href="/transaction/payterm/detail/{{$u->id_transaction}}" target="_blank" style="width: inherit;">
                              <button class="btn btn-primary w-100 fw-bold">Pay Term</button>
                          </a>
                        @endif

                      {{-- Member Inactive --}}
                        @if ($member->istatus_member == 0)
                          <a href= '#' style="width: inherit;">
                              <button class="btn btn-primary w-100 fw-bold" disabled >Pay Term</button>
                          </a>
                        @endif
                        @endif
                    </div>
```

Gambar 3. 88 HTML Transaction_payment

Pada tampilan HTML transaction_payment, *developer* melakukan penambahan kondisi IF jika *variable* member bernilai *null* maka button *payterm* akan *disable* begitu juga jika *variable* member ada nilai tapi status member bernilai 0 maka button payterm akan *disable*. Button *payterm* berfungsi ketika user telah menjadi member dan berstatus *active* atau bernilai 1. Pada status member, *developer* juga membuat sebuah *hypertext reference* /transaction/payterm/detail/{{$u->id_transaction}} seperti gambar 3.88 untuk bisa berpindah ke tampilan data *payterm* sesuai dengan id transaction.



Gambar 3. 89 Tampilan Payterm

Hasil tampilan button *payterm* dapat terlihat pada gambar 3.89 yang terlihat terdapat button disable dan button yang tidak disable

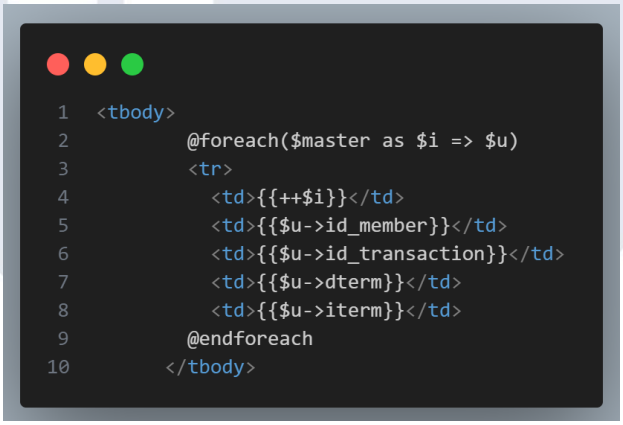
```php
public function index_headerterm($id)
    {
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        $master = DB::table('tmember_dtl')
            ->join('tmember', 'tmember_dtl.id_member', '=', 'tmember.id_member')
            ->join('ttransaction_hdr', 'tmember_dtl.id_transaction', '=', 'ttransaction_hdr.id_transaction')
            ->where('ttransaction_hdr.id_transaction', $id)
            ->select('tmember_dtl.*', 'tmember.id_member', 'ttransaction_hdr.id_transaction')->get();
        //dd($master);

        return view('member_detail', compact('master', 'nama_web'));
    }
```

Gambar 3. 90 Function untuk data Payterm

Setelah membuat button *payterm* maka *developer* melanjutkan untuk memanggil data *payterm* yang telah dimasukkan kedalam table member_dtl dengan membuat sebuah *function* baru yang bernama index_headerterm. Dalam *function* terebut terdapat sebuah variabel master yang memanggil data dari table member_dtl dan mealkukan join ke beberapa table seperti member untuk mendapatkan id_member dan transaction_hdr untuk mendapatkan id_transaction dengan kondosi id_transaction sesuai dengan id transaction dari request user. *Variable* master ini menggunakan *method* GET. Dari variabel master ini akan dipassing ke tampilan member_detail seperti gambar 3.90.

Gambar 3. 91 Route tampilan data Payterm

Setelah itu *developer* membuat sebuah *route* dari *hypertext reference* yang berada di transaction_payment agar bisa menghubungkan *function* dan tampilan member detail seperti gambar 3.91.



Gambar 3. 92 HTML Member_detail

Berikutnya pada tampilan member_detail ini developer melakukan *foreach* karena menggunakan *method* GET sehingga diperlukan *foreach* agar data bisa tampil semua ke tampilan member_detail. Data yang ditampilkan yaitu id_member, id_transaction, tanggal pembayaran dan harga dari term tersebut seperti gambar 3.92.

8) Pengembangan Transaction dalam Menjumlahkan Point

a) 18 Sepetmber – 23 September

```php
if (is_null($member)) {
            $tsetting7 = DB::table('tglobalsetting')->where('vname', 'calc_point')->first();
            $min_point = $tsetting7->ivalue;
            $total_point = 0;

            DB::table('ttransaction_hdr')->insert([
                'id_user' => Auth::user()->id_user,
                'id_payment' => $check_payment->id_payment,
                'itotal_quantity' => $count_data,
                'itracking_status' => 0,
                'vaddress' => $request->address,
                'vnote' => $request->notes,
                'itotal_price' => $request->amount,
                'itotal_point' => $total_point,
                'vcrea' => Auth::user()->email,
                'dcrea' => Carbon::now(),
            ]);
        }
```

```php
elseif (!is_null($member)) {
            if ($total_price >= $min_purchase) {
                if ($member->istatus_member == 1) {
                    $tsetting7 = DB::table('tglobalsetting')->where('vname', 'calc_point')->first();
                    $min_point = $tsetting7->ivalue;
                    $total_point = $total_price / $min_point;
                    //dd($total_point);

                    DB::table('ttransaction_hdr')->insert([
                        'id_user' => Auth::user()->id_user,
                        'id_payment' => $check_payment->id_payment,
                        'itotal_quantity' => $count_data,
                        'itracking_status' => 0,
                        'vaddress' => $request->address,
                        'vnote' => $request->notes,
                        'itotal_price' => $request->amount,
                        'itotal_point' => $total_point,
                        'vcrea' => Auth::user()->email,
                        'dcrea' => Carbon::now(),
                    ]);
                } elseif ($member->istatus_member == 0) {
                    $tsetting7 = DB::table('tglobalsetting')->where('vname', 'calc_point')->first();
                    $min_point = $tsetting7->ivalue;
                    $total_point = 0;

                    DB::table('ttransaction_hdr')->insert([
                        'id_user' => Auth::user()->id_user,
                        'id_payment' => $check_payment->id_payment,
                        'itotal_quantity' => $count_data,
                        'itracking_status' => 0,
                        'vaddress' => $request->address,
                        'vnote' => $request->notes,
                        'itotal_price' => $request->amount,
                        'itotal_point' => $total_point,
                        'vcrea' => Auth::user()->email,
                        'dcrea' => Carbon::now(),
                    ]);
                }
            } elseif ($total_price <= $min_purchase) {
                $tsetting7 = DB::table('tglobalsetting')->where('vname', 'calc_point')->first();
                $min_point = $tsetting7->ivalue;
                $total_point = 0;

                DB::table('ttransaction_hdr')->insert([
                    'id_user' => Auth::user()->id_user,
                    'id_payment' => $check_payment->id_payment,
                    'itotal_quantity' => $count_data,
                    'itracking_status' => 0,
                    'vaddress' => $request->address,
                    'vnote' => $request->notes,
                    'itotal_price' => $request->amount,
                    'itotal_point' => $total_point,
                    'vcrea' => Auth::user()->email,
                    'dcrea' => Carbon::now(),
                ]);
            }
        }

```

Gambar 3. 93 Code Point Member di Xendit Controller

Pada kegiatan ini, *developer* ditugaskan untuk menjumlahkan point yang telah didapatkan oleh users. Oleh karena itu untuk bisa mendapatkan point terlebih dahulu melakukan penambahan beberapa code di *Xendit controller* ketika menambahkan data ke table

transaction_hdr. Variabel yang dibutuhkan yaitu variabel member, min_purchase, dan total price yang telah dibuat sebelumnya. Setelah itu dengan kondisi IF untuk memeriksa apakah user tersebut termasuk member atau tidak dari variabel member jika bernilai *null* maka user tersebut akan mendapatkan total point sebesar nilai 0 dan nilai tersebut yang akan dimasukan ke dalam tabel transaction_hdr dengan perintah DML (*Data Manipulation Language*) yaitu *insert*.

Namun, jika member tidak bernilai *null* dan total_price telah memenuhi minimal pembelian serta status member bernilai 1 maka akan mendapatkan point sebesar totalprice dibagi dengan min_point dari tabel global setting. Jika statu member bernilai 0 maka di kolom itotal_point pada transaction_hdr juga bernilai 0. Begitu juga dengan total price yang tidak memenuhi minimal pembelian dapat terlihat pada gambar 3.93.

```
// Point
    $user_point = DB::table('ttransaction_hdr')
        ->where('id_user', Auth::user()->id_user)
        ->get(['itotal_point']);
    //dd($user_point);

    $member = DB::table('tmember')->where('id_user', Auth::user()->id_user)->first();
    $point = [];
    //dd($point);

    $count_point = 0;

    foreach ($user_point as $i => $u) {
        $point[$i]['index'] = $i;
        $point[$i]['itotal_point'] = $u->itotal_point;
    }
    foreach ($point as $count) {
        DB::table('tmember')
            ->where('id_user', Auth::user()->id_user)
            ->where('istatus_member', 1)->update([
                'ipoint' =>  $count_point += $count['itotal_point'],
            ]);
    }
```

Gambar 3. 94 Code untuk menampilkan point di tampilan Home (Dashbroad Home)

Setelah mendapatkan point dari setiap transaski yang dilakukan oleh user maka berikutnya *developer* perlu menjumlahkan semua point dari table transaction_hdr kedalam table member dan menampilkan kedalam tampilan home dan profile setting. Oleh karena itu, perlu penambahan

pada *function* dashboard_home di bagian *login controller* dan *function* usersetting di bagian *user controller*. Pertama-tama, *developer* melakukan penambahan pada dashboard_home berupa variabel user_point yang memanggil data dari tabel transaction_hdr dengan kondisi where id_user sesuai dengan id_user yang lagi login ke web *e-commere* dan menggunakan *method* GET untuk mengambil semua data dari kolom itotal_point.

Developer juga membuat sebuah *variable* member untuk memanggil data member dan membuat variabel untuk bisa menampung jumlah point user. Setalah itu, *developer* melakukan *foreach* untuk bisa memanggil data dari variable user_point dan memilih kebutuhan *developer* yaitu itotal_point. Berikutnya *developer* melakukan *foreach* sekali lagi agar semua data dari itotal_point bisa dimasukan ke dalam table member dengan 2 kondisi where yaitu id_user harus sesuai dengan login user dan status member bernilai 1.

Dalam menambahkan data ke tabel member tidak menggunakan perintah *insert* karena point akan elaalu berubah maka dari itu menggunakan perintah DML (*Data Manipulation Language*) yaitu *update*. Dalam mengupdate ini menggunakan variabel count_point sebagai dasar untuk diperbarui dan akan dijumlah dari itotal_point seperti gambar 3.94.

```php
public function usersettings()
    {
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        $user_point = DB::table('ttransaction_hdr')
            ->where('id_user', Auth::user()->id_user)
            ->get(['itotal_point']);
        //dd($user_point);
        $member = DB::table('tmember')->where('id_user', Auth::user()->id_user)->first();
        $count_point = 0;
        $point = [];
        foreach ($user_point as $i => $u) {
            $point[$i]['index'] = $i;
            $point[$i]['itotal_point'] = $u->itotal_point;
        }
        foreach ($point as $count) {
            DB::table('tmember')
                ->where('id_user', Auth::user()->id_user)
                ->where('istatus_member', 1)->update([
                    'ipoint' =>  $count_point += $count['itotal_point'],
                ]);
        }
        return view('profile_settings', compact('point', 'member', 'nama_web'));
    }
```

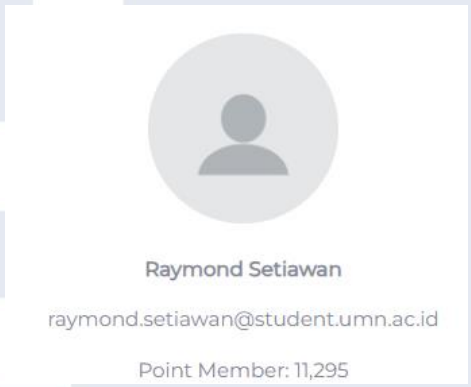
Gambar 3. 95 Code untuk menampilkan point di tampilan profile setting (User Controller)

Gambar 3.95 merupakan code dari profile setting. Pada bagian usersetting dilakukan hal yang sama dengan dashboard_home untuk menampilkan point di bagian profile setting

```
<img
            src="{{asset('assets/picture').'/'.(Auth::user()->profile_picture)}}"
            alt="Foto Profil"
            style="border-radius: 50%; width:150px; height:150px;"
            class="img-fluid mt-3 "/>
            <br><br>
            </a>
          <p class="fw-bold">{{ Auth::user()->vname }}</p>
          <p hidden class="fw-bold">{{ Auth::user()->id_user }}</p>
          <p styeclass="pb-4">{{ Auth::user()->email }}</p>
          @if (is_null($member))
            <input
              name="point"
              type="number"
              class="form-control"
              value="{{0}}"
              aria-label="Sizing example input"
              aria-describedby="inputGroup-sizing-default"
              hidden/>
              {{number_format(0)}}
          @endif

          @if (!is_null($member))
            {{-- Member Active --}}
              @if ($member->istatus_member == 1)
                <input
                  name="point"
                  type="number"
                  class="form-control"
                  value="{{$member->ipoint}}"
                  aria-label="Sizing example input"
                  aria-describedby="inputGroup-sizing-default"
                  hidden/>
                Point Member: {{number_format($member->ipoint)}}
              @endif

            {{-- Member Inactive --}}
              @if ($member->istatus_member == 0)
                <input
                  name="point"
                  type="number"
                  class="form-control"
                  value="{{0}}"
                  aria-label="Sizing example input"
                  aria-describedby="inputGroup-sizing-default"
                  hidden/>
                  {{number_format(0)}}
              @endif
          @endif
```

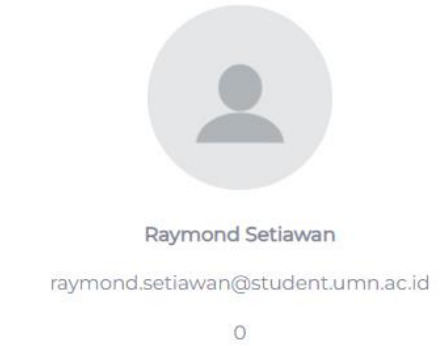
Gambar 3. 96 HTML di profile setting

Pada HTML ini developer melakukan penambahan 3 kondisi if yaitu dari variabel member jika bernilai *null* maka tampilan point pada profile setting akan bernilai 0. Namun, jika member tidak bernilai null dan memiliki status member yang bernilai 1 maka point akan ditampilkan dari variabel member mengambil value ipoint



Gambar 3. 97 Tampilan Profile setting ketika menjadi Member

Pada gambar 3.97 merupakan hassil ketika user mendapatkan point dan telah menjadi member maka total point yang dikumpulkan dapat terlihat.



Gambar 3. 98 Tampilan Profile setting ketika tidak menjadi Member

Namun, jika status member bernilai 0 maka tampilan point pada profile setting juga akan berubah menjadi 0 seperti pada gambar 3.98.

9) Perbaikan Point menjadi Point per Item

a) 25 Sepetmber – 26 September

```php
$member_term = DB::table('tmember')
            ->where('id_user', Auth::user()->id_user)
            ->where('istatus_member', 1)->first();

        $tsetting8 = DB::table('tglobalsetting')->where('vname', 'min_term')->first();
        $min_term = $tsetting8->ivalue;
        $lop = 0;

        $check_transaction = DB::table('ttransaction_hdr')->where('dcrea', Carbon::now())->first();

        if (!is_null($check_transaction)) {
            //Non Member
            if (is_null($member)) {
                foreach ($cart_data as $i) {
                    DB::table('titem_hdr')->where('id_item', $i->id_item);
                    DB::table('titem_hdr')->where('id_item', $i->id_item)->decrement('istock', $i->iquantity);
                    DB::table('ttransaction_dtl')->insert([
                        'id_transaction' => $check_transaction->id_transaction,
                        'id_item' => $i->id_item,
                        'iquantity' => $i->iquantity,
                        'iprice' => ($i->iprice * $i->iquantity),
                        'ipoint' => 0,
                        'dspack' => Carbon::now(),
                        'vcrea' => Auth::user()->email,
                        'dcrea' => Carbon::now(),
                    ]);
                }
                DB::table('tcart')->where('id_user', Auth::user()->id_user)->delete();
            }
            // Member
            if (!is_null($member)) {
                if ($total_price >= $min_purchase) {
                    if ($member->istatus_member == 1) {
                        foreach ($cart_data as $i) {
                            DB::table('titem_hdr')->where('id_item', $i->id_item)->first();
                            DB::table('titem_hdr')->where('id_item', $i->id_item)->decrement('istock', $i->iquantity);
                            DB::table('ttransaction_dtl')->insert([
                                'id_transaction' => $check_transaction->id_transaction,
                                'id_item' => $i->id_item,
                                'iquantity' => $i->iquantity,
                                'iprice' => ($i->iprice * $i->iquantity),
                                'ipoint' => ($i->iprice * $i->iquantity) / $min_point,
                                'dspack' => Carbon::now(),
                                'vcrea' => Auth::user()->email,
                                'dcrea' => Carbon::now(),
                            ]);
                        }
                        for ($lop = 1; $lop <= $min_term; $lop++) {
                            DB::table('tmember_dtl')->insert([
                                'id_member' => $member_term->id_member,
                                'id_transaction' => $check_transaction->id_transaction,
                                'iterm' => $check_transaction->itotal_price / $min_term,
                                'vcrea' => Auth::user()->email,
                                'dcrea' => Carbon::now(),
                            ]);
                        }
                        DB::table('tcart')->where('id_user', Auth::user()->id_user)->delete();
                    }
                } elseif ($member->istatus_member == 0) {
                    foreach ($cart_data as $i) {
                        DB::table('titem_hdr')->where('id_item', $i->id_item);
                        DB::table('titem_hdr')->where('id_item', $i->id_item)->decrement('istock', $i->iquantity);
                        DB::table('ttransaction_dtl')->insert([
                            'id_transaction' => $check_transaction->id_transaction,
                            'id_item' => $i->id_item,
                            'iquantity' => $i->iquantity,
                            'iprice' => ($i->iprice * $i->iquantity),
                            'ipoint' => 0,
                            'dspack' => Carbon::now(),
                            'vcrea' => Auth::user()->email,
                            'dcrea' => Carbon::now(),
                        ]);
                    }
                    DB::table('tcart')->where('id_user', Auth::user()->id_user)->delete();
                }
            }
            if ($total_price <= $min_purchase) {
                foreach ($cart_data as $i) {
                    DB::table('titem_hdr')->where('id_item', $i->id_item);
                    DB::table('titem_hdr')->where('id_item', $i->id_item)->decrement('istock', $i->iquantity);
                    DB::table('ttransaction_dtl')->insert([
                        'id_transaction' => $check_transaction->id_transaction,
                        'id_item' => $i->id_item,
                        'iquantity' => $i->iquantity,
                        'iprice' => ($i->iprice * $i->iquantity),
                        'ipoint' => 0,
                        'dspack' => Carbon::now(),
                        'vcrea' => Auth::user()->email,
                        'dcrea' => Carbon::now(),
                    ]);
                }
                DB::table('tcart')->where('id_user', Auth::user()->id_user)->delete();
            }
```

Gambar 3. 99 Perbaikan code point menjadi point per Item

Pada kegiataan ini, *developer* diberikan tugas untuk melakukan penambahan pada point agar total point dari item yang di transaction_hdr bisa dijabarkan ke dalam table transaction_dtl sesuai item. Oleh karena itu, *developer* melakukan penambahan pada 1 baris code yang telah dibuatkan ketika melakukan *payterm*. Penambahan code ini berada di *foreach* cart_data ketika melakukan *insert* data ke dalam tabel transaction_dtl karena foreach ini memasukan data ke transaction _dtl sesuai item yang dibeli oleh user.

Untuk point per item ini memiliki kondisi yang sama dengan point yang berada di transaction_hdr yaitu user akan mendapatkan point ketika menjadi member, total pembelian telah memenuhi minimal pembelian dan memiliki status bernilai 1. Jika tidak memenihi syarat tersebut maka point yang didapatkan bernilai 0. Code yang ditambahkan yaitu 'ipoint' => ($i->iprice * $i->iquantity) / $min_point sehingga harga item tersebut akan dikalikan terlebih dahulu dengan quantity yang dibeli user yang nanti akan dibagi dengan variabel min_point dapat terlihat pada gambar 3.99.

id_transaction	id_payment	id_user	itotal_quantity	itracking_status	itotal_price	itotal_point
162	183	55,583	2	3	1,860,000	1,980

Gambar 3. 100 Database pada table transaction_hdr

Pada gambar 3.100 merupakan isi dari table transaction_hdr yang menampung point dari user setiap melakukan transaksi.

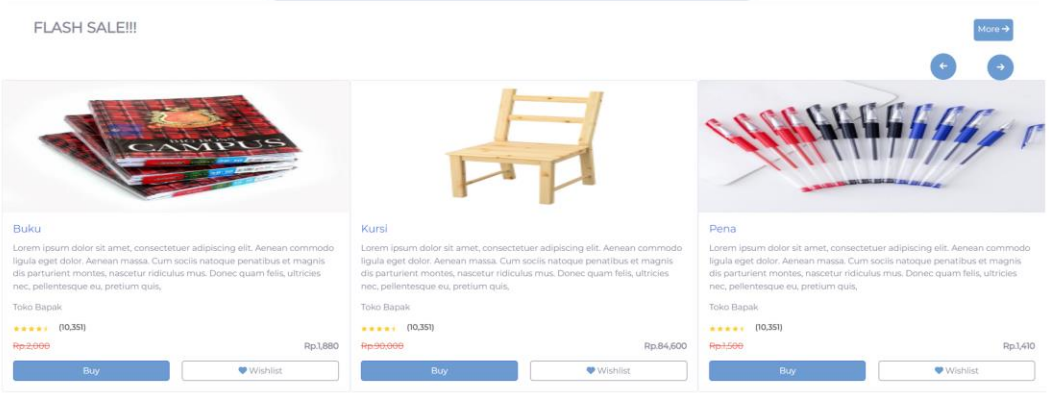
id_transactiondtl	id_transaction	id_item	iquantity	iprice	ipoint
203	162	333,346	9	1,080,000	1,080
204	162	333,344	9	900,000	900

Gambar 3. 101 Database pada table transaction_dtl

Hasil dari perubahan ini dapat terlihat pada gambar 3.101 yang setiap item memiliki point nya tersendiri dan jika dijumlahkan maka total jumlahnya akan sama dengan table transction_hdr.

10) Perbaikan Tampilan Home (Slick Slide)

a) 27 Sepetmber – 29 September



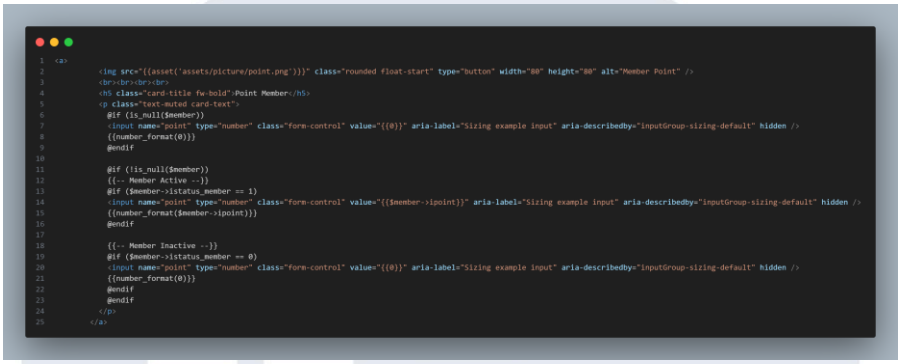
Gambar 3. 102 Tampilan flashsale setelah menjadi slick slide

Pada kegiatan ini, *developer* dan *front-end developer* memperbaiki tampilan home menjadi *slick slide*. *Front-end* telah mempersiapkan tampilan *slick slide*. *Developer* pada kegiatan ini membantu merapikan setelah tampilan home berubah menjadi *slick slide* karena ini berhubungan *function* dashboard_home dapat terlihat pada gambar 3.102.

Gambar 3. 103 HTML Home untuk Saldo Anda

Tidak hanya itu, *developer* dan *front-end developer* juga memperbaiki tampilan home untuk bisa menampilkan beberapa fitur yang telah ditambahkan seperti saldo anda, point member, *payterm*, *return product*. Gambar 3.103 merupakan HTML untuk saldo Anda dan tampilan tersebut menggunakan kondisi jika user telah menjadi member maka akan muncul saldo yang telah didapatkan jika user belum menjadi

member atau status member tidak *active* maka dan saldo akan bernilai 0.



```
<a>
        <img src="{{asset('assets/picture/point.png')}}" class="rounded float-start" type="button" width="80" height="80" alt="Member Point" />
        <br><br><br><br>
        <h5 class="card-title fw-bold">Point Member</h5>
        <p class="text-muted card-text">
          @if (is_null($member))
          <input name="point" type="number" class="form-control" value="{{0}}" aria-label="Sizing example input" aria-describedby="inputGroup-sizing-default" hidden />
          {{number_format(0)}}
          @endif

          @if (!is_null($member))
          {{-- Member Active --}}
          @if ($member->istatus_member == 1)
          <input name="point" type="number" class="form-control" value="{{$member->ipoint}}" aria-label="Sizing example input" aria-describedby="inputGroup-sizing-default" hidden />
          {{number_format($member->ipoint)}}
          @endif

          {{-- Member Inactive --}}
          @if ($member->istatus_member == 0)
          <input name="point" type="number" class="form-control" value="{{0}}" aria-label="Sizing example input" aria-describedby="inputGroup-sizing-default" hidden />
          {{number_format(0)}}
          @endif
          @endif
        </p>
      </a>
```

Gambar 3. 104 HTML Home untuk Point Member

Pada gambar 3.104 merupakan HTML untuk Point Member. Tampilan tersebut menggunakan kondisi jika user telah menjadi member maka akan muncul point yang telah didapatkan jika user belum menjadi member atau status member tidak *active* maka point member akan bernilai 0.

```
@if (is_null($member))
      <a style="text-decoration:none" href="#">
        <img src="{{asset('assets/picture/paylater.png')}}" class="rounded float-start" type="button" width="80" height="80" alt="Pay Later" />
        <br><br><br><br>
        <h5 class="card-title fw-bold">Pay term</h5>
      </a>
      @endif
      @if (!is_null($member))
      {{-- Member Active --}}
      @if ($member->istatus_member == 1)
      <a style="text-decoration:none" href="/transaction_user/payment">
        <img src="{{asset('assets/picture/paylater.png')}}" class="rounded float-start" type="button" width="80" height="80" alt="Pay Later" />
        <br><br><br><br>
        <h5 class="card-title fw-bold">Pay term</h5>
      </a>
      @endif

      {{-- Member Inactive --}}
      @if ($member->istatus_member == 0)
      <a style="text-decoration:none" href="#">
        <img src="{{asset('assets/picture/paylater.png')}}" class="rounded float-start" type="button" width="80" height="80" alt="Pay Later" />
        <br><br><br><br>
        <h5 class="card-title fw-bold">Pay term</h5>
      </a>
      @endif
      @endif
```

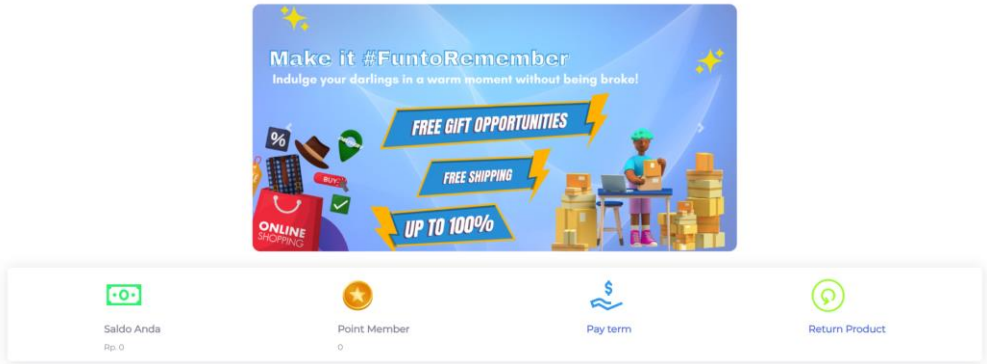
Gambar 3. 105 HTML Home untuk Payterm

Sementara gambar 3.105 merupakan HTML untuk *payterm*. Fitur ini menggunakan kondisi ketika users telah menjadi member maka baru bisa mengklik fitur tersebut jika belum menjadi member atau status tidak *active* maka tidak bisa mengklik fitur tersebut.

```
@if (is_null($member))
        <a style="text-decoration:none" href="#">
          <img src="{{asset('assets/picture/return.png')}}" class="rounded float-start" type="button" width="80" height="80" alt="Pay Later" />
          <br><br><br><br>
          <h5 class="card-title fw-bold">Return Product</h5>
        </a>
        @endif
        @if (!is_null($member))
        {{-- Member Active --}}
        @if ($member->istatus_member == 1)
        <a style="text-decoration:none" href="return_transaction">
          <img src="{{asset('assets/picture/return.png')}}" class="rounded float-start" type="button" width="80" height="80" alt="Pay Later" />
          <br><br><br><br>
          <h5 class="card-title fw-bold">Return Product</h5>
        </a>
        @endif

        {{-- Member Inactive --}}
        @if ($member->istatus_member == 0)
        <a style="text-decoration:none" href="#">
          <img src="{{asset('assets/picture/return.png')}}" class="rounded float-start" type="button" width="80" height="80" alt="Pay Later" />
          <br><br><br><br>
          <h5 class="card-title fw-bold">Return Product</h5>
        </a>
        @endif
        @endif
```

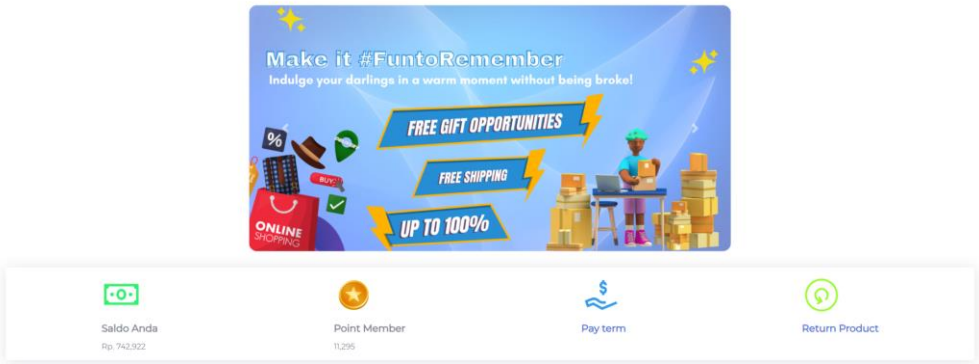
Gambar 3. 106 HTML Home untuk Return Product

Sementara gambar 3.106 HTML untuk *return product*. Fitur ini menggunakan kondisi ketika users telah menjadi member maka baru bisa mengklik fitur tersebut jika belum menjadi member atau status tidak *active* maka tidak bisa mengklik fitur tersebut.



Gambar 3. 107 Tampilan Home ketika user tidak menjadi member

Hasil tampilan bisa terlihat seperti pada gambar 3.107 yang menampilkan bagi users yang belum menjadi member sehingga point dan saldo masih bernilai 0 dan fitur return product dan payterm belum bisa di klik.



Gambar 3. 108 Tampilan Home ketika user menjadi member

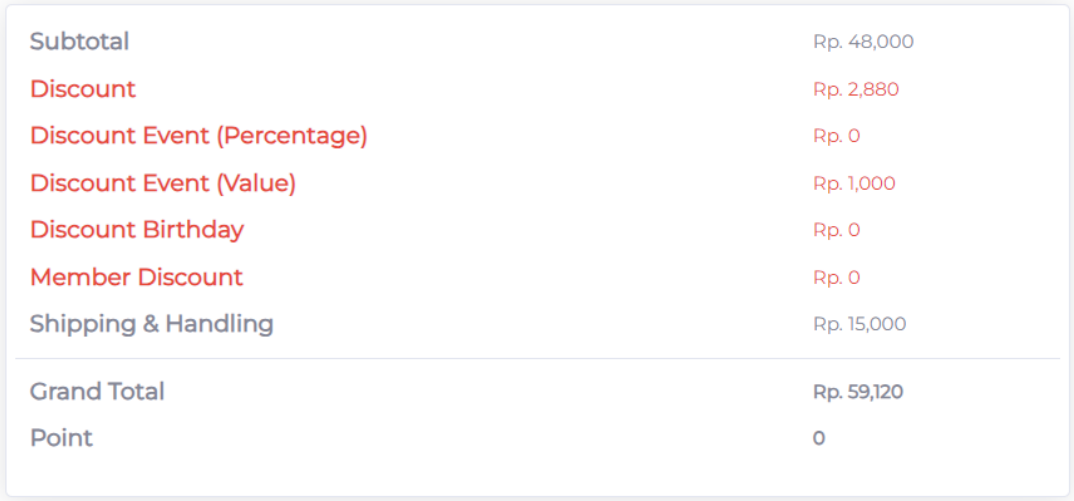
Hasil tampilan bisa terlihat seperti pada gambar 3.108 yang menampilkan bagi users yang telah menjadi member sehingga point dan saldo terdapat nilainya dan fitur return product dan payterm bisa di klik.

11) Perbaikan Tampilan Cart

a) 30 Sepetmber – 30 September

Gambar 3. 109 Tampilan Cart yang belum dirapikan

Pada kegiatan ini, *developer* melakukan merapikan tampilan cart yang tidak rapi seperti pada gambar 3.109 yang masih berantakan dan belum sejajar.



Gambar 3. 110 Tampilan Cart yang telah dirapikan

Pada gambar 3.110 merupakan hasil dari tampilan cart yang telah rapi dengan cara merapikan setiap DIV classnya di HTML cart dan disesuaikan kembali sehingga bisa menjadi tampilan cart yang rapi.

12) Pengembangan Return item I (Admin)

a) 2 Oktober – 2 Oktober

Pada kegiatan ini *developer* membantu teman *developer* selaku *back-end developer* dalam melakukan return item pada sisi admin. *Developer* membantu dalam memecahkan masalah atau problem yang dihadapi oleh teman *developer* yaitu tampilan ketika *accept* dan *reject* serta *route*. *Developer* tersebut kendala ketika *accept* dan *reject* dijadikan dalam *pop-up modal*. *Developer* mengatasi problem tersebut dengan membuat tampilan baru untuk *accept* dan *reject*

```
@extends('template')
@section('content')
<div class="card-body">
    <form action="{{url('master/retur/decline')}}" method="post" enctype="multipart/form-data">
        @csrf

        <input type="text" name="id" value="{{$retur->id_transactiondtl}}" hidden>
        <div class="form-group">
            <label for="input-group mb-3 mt-3" class="form-label">Reason Admin to decline the Item</label>
            <input name="reasonadmin" type="text" class="form-control" placeholder="Enter Your Reason" required>
        </div>
        <div class="form-group">
            <button type="approval_status" class="btn btn-primary">Yes</button>
            <button type="button" onclick="history.back();" class="btn btn-primary">Back</button>
        </div>
    </form>
</div>

@endsection
```

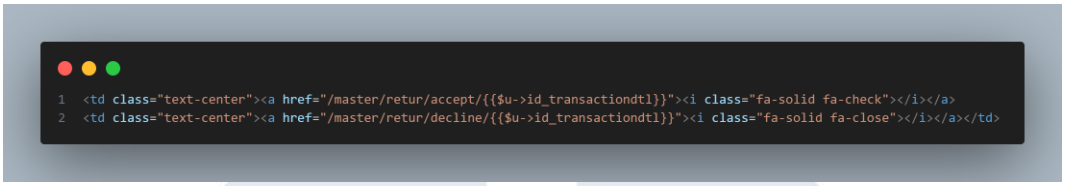
Gambar 3. 111 HTML Master Return Decline

Pada gambar 3.111 merupakan HTML untuk decline return product yang berfungsi untuk mendecline return product sesuai dengan id_transaction yang ingin di decline.

```
@extends('template')
@section('content')
<div class="card-body">
    <form action="{{url('/master/return/accept')}}" method="post" enctype="multipart/form-data">
        @csrf
        <input type="text" name="id" value="{{$retur->id_transactiondtl}}" hidden>
        <div class="form-group">
            <label class="me-3" for="">Are you sure accept this return?</label>
        </div>
        <button type="approval_status" class="btn btn-primary">Yes</button>
        <button type="button" onclick="history.back();" class="btn btn-primary">Back</button></a>
    </form>
</div>
@endsection
```

Gambar 3. 112 HTML Master Return Accept

Pada gambar 3.112 merupakan HTML untuk accept return product yang berfungsi untuk menaccept return product sesuai dengan id_transaction yang ingin di accept.



Gambar 3. 113 HTML Master Return

Sementara pada HTML master return juga melakukan perubahan seperti pada gambar 3.113 yang *hypertext reference* dari icon tersebut terdapat URL dari kedua *tampilan* accept dan *decline* yang berisikan value id_transactiondtl.

Gambar 3. 114 Route Master Return

Developer juga memperbaiki *route* yang telah dibuat oleh teman *developer* selaku *back-end developer* tersebut dapat terlihat pada gambar 3.114 untuk dapat berpindah tampilan ke *accept* atau *decline* ini diperlukan *route* dengan *method* GET karena *method* ini akan mengambil terlebih dulu id dari data yang dipilih oleh admin dan berpindah tampilan accept atau decline sesuai pilihan admin. Ketika admin *accept* atau *decline* dari return product maka akan menggunakan *method* POST sesuai *method* di form yang terdapat di HTML *decline* atau *accept*.

13) Perbaikan Transaction Discount Event per item

a) 2 Oktober – 3 Oktober

```php
$discevent = DB::table('ttransaction_event')
            ->join('tdiscount', 'ttransaction_event.id_discount', '=', 'tdiscount.id_discount')
            ->join('tevent', 'ttransaction_event.id_event', '=', 'tevent.id_event')
            ->join('titem_hdr', 'ttransaction_event.id_item', '=', 'titem_hdr.id_item')
            ->join('users', 'ttransaction_event.id_user', '=', 'users.id_user')
            ->select('ttransaction_event.*', 'tdiscount.*', 'tevent.*', 'titem_hdr.*', 'users.vname')
            ->where('status', 1)->first();

        $discevent2 = DB::table('ttransaction_event')
            ->join('tdiscount', 'ttransaction_event.id_discount', '=', 'tdiscount.id_discount')
            ->join('tevent', 'ttransaction_event.id_event', '=', 'tevent.id_event')
            ->join('titem_hdr', 'ttransaction_event.id_item', '=', 'titem_hdr.id_item')
            ->join('users', 'ttransaction_event.id_user', '=', 'users.id_user')
            ->where('tevent.dsendevent', '>=', Carbon::today())
            ->select('ttransaction_event.*', 'tdiscount.*', 'tevent.*', 'titem_hdr.*', 'users.vname')
            ->get();

        if ($discevent != null) {
            $total_disc_value = $discevent->value;
            $total_disc_percentage = $discevent->percentage;
        } else {
            $total_disc_value = 0;
            $total_disc_percentage = 0;
        }
```

Gambar 3. 115 Code Perbaikan Discount Event per Item

Pada kegiatan ini, *developer* memperbaiki transaction discount event yang telah pernah dibuat sebelumnya agar *discount* event ini bisa disesuaikan dengan item barang yang *discount*. *Developer* juga menambahkan kondisi IF ketika data dari variabel discevent tidak bernilai *null* maka mengambil value dari variabel discevent tapi jika bernilai *null* maka variabel total_disc_value dan total_disc_percentage akan bernilai 0 dapat terlihat pada gambar 3.115.

```
foreach ($count as $i => $u) {
            $item[$i]['index'] = $i;

            $vcategory = DB::table('tcategory')->where('id_category', $u->id_category)->first();
            $picture = DB::table('titem_dtl')->where('id_item', $u->id_item)->select('picture')->orderBy('id_itemdtl', 'asc')->first();

            if ($discevent != null) {
                $discount = DB::table('ttransaction_event')
                    ->join('tcart', 'ttransaction_event.id_item', '=', 'tcart.id_item')
                    ->join('titem_hdr', 'tcart.id_item', '=', 'titem_hdr.id_item')
                    ->where('ttransaction_event.id_event', $discevent->id_event)
                    ->get();
                //dd($discevent);
                //dd($price);

                $item_discount = [];
                foreach ($discount as $y => $x) {
                    $item_discount[$y]['id_item'] = $x->id_item;
                    $item_discount[$y]['iprice'] = $x->iprice;
                }
                //dd($item_discount);

                $item_cart = [];
                foreach ($count as $z => $t) {
                    $item_cart[$z]['id_item'] = $t->id_item;
                    $item_cart[$z]['iprice'] = $t->iprice;
                }

                if ($item_discount == null) {
                    $totalpercentage = 0;
                } else {
                    $totalpercentage = ($x->iprice * $total_disc_percentage / 100);
                }

                if ($item_discount == null) {
                    $totalvalue = 0;
                } else {
                    $totalvalue = ($total_disc_value);
                }

                if ($picture) {
                    $item[$i]['picture'] = $picture->picture;
                } else {
                    $item[$i]['picture'] = null;
                }

                $item[$i]['id'] = $u->id_item;
                $item[$i]['vname_item'] = $u->vname_item;
                $item[$i]['vname'] = $u->vname;
                $item[$i]['vcategory'] = $vcategory->vcategory;
                $item[$i]['iquantity'] = $u->iquantity;
                $item[$i]['iprice_after'] = $u->iprice - ($u->iprice * $disc_flashsale);
                $item[$i]['iprice_eventpercentage'] = ($totalpercentage);
                $item[$i]['iprice_eventvalue'] = ($totalvalue);
                $item[$i]['iprice'] = $u->iprice;
                $item[$i]['iactive'] = $u->iactive;
            }
```

Gambar 3. 116 Code Perbaikan Discount Event per Item (I)

Setelah itu, pada bagian *foreach* terdapat kondisi IF jika disevent tidak bernilai *null* maka perhitungan *discount* event bisa digunakan sesuai event yang nanti dipilih oleh user seperti gamabr 3.116. Didalam IF tersebut, *developer* menambahkan variabel *discount* yang memanggil data dari table transaction_event dengan melakukan joVin ke beberapa *table* seperti *table* cart dan *table* item_hdr dan menggunakan kondisi where id_event dari variabel discevent.

Berikutnya melakukan *foreach* terhadap varibel *discount* dan *developer* hanya membutuhkan id_item dan harga item. Selain itu

developer juga melakukan *foreach* untuk variabel count untuk mengambil id_item dan harga item. Selanjutnya, *developer* menambahkan 2 kondisi IF untuk variabel item_discount yang berisikan *array* dari *discount*. Jika item_discount bernilai *null* maka variabel dari totalpercentage bernilai 0 tapi jika tidak *null* maka variabel totalpercentage bernilai harga barang *dari* table transaction_event yang dikali dengan variabel total_disc_percentage / 100. Begitu juga dengan *discount* yang bernilai value. Jika item discount tidak *null* maka variabel total_value akan bernilai sesuai variabel total_disc_value.

Developer menggantikan 2 baris yang pernah developer tambahkan. Dua baris *array* yang sebelumnya seperti dibawah ini:

- $item[$i]['iprice_eventpercentage'] = $u->iprice - ($price->iprice * $total_disc_percentage/100);
- $item[$i]['iprice_eventvalue'] = $u->iprice - ($price->iprice - $total_disc_value);

Developer menggantikan 2 baris *array* tersebut menjadi lebih sederhana seperti ini:

- $item[$i]['iprice_eventpercentage'] = ($totalpercentage);
- $item[$i]['iprice_eventvalue'] = ($totalvalue);

```
else {

                $totalpercentage = 0;
                $totalvalue = 0;

                if ($picture) {
                    $item[$i]['picture'] = $picture->picture;
                } else {
                    $item[$i]['picture'] = null;
                }

                $item[$i]['id'] = $u->id_item;
                $item[$i]['vname_item'] = $u->vname_item;
                $item[$i]['vname'] = $u->vname;
                $item[$i]['vcategory'] = $vcategory->vcategory;
                $item[$i]['iquantity'] = $u->iquantity;
                $item[$i]['iprice_after'] = $u->iprice - ($u->iprice * $disc_flashsale);
                $item[$i]['iprice_eventpercentage'] = ($totalpercentage);
                $item[$i]['iprice_eventvalue'] = ($totalvalue);
                $item[$i]['iprice'] = $u->iprice;
                $item[$i]['iactive'] = $u->iactive;
            }

            //dd($item);
        }
```

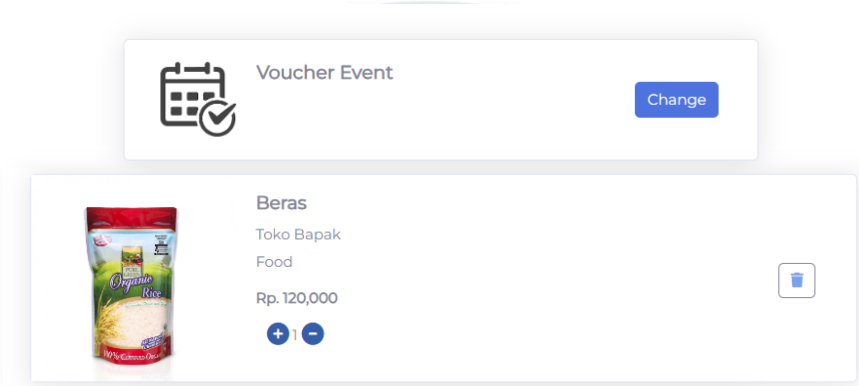
Gambar 3. 117 Code Perbaikan Discount Event per Item (II)

Gambar 3.117 ini adalah kondisi ELSE yang berarti jika disecvent bernilai *null* maka variable totalpercentage akan bernilai 0 dan totalvalue juga bernilai 0.

```
<div class="col-md-8 align-items-center">
          <h5 class="card-title fw-bold">Voucher Event</h5>
          <p class="text-muted card-text">
            @if($discevent != null)
            <input name="discevent" type="text" class="form-control" value="{{$discevent->vdesc}}" aria-label="Sizing example input" aria-describedby="inputGroup-sizing-default" hidden />
            {{$discevent->vname}}
            <br>
            {{$discevent->vname_item}}
            <br>
            {{$discevent->vdesc}}
            <br>{{$discevent->dstartevent}} - {{$discevent->dsendevent}}
          </p>
          @endif
          @if($discevent == null)
          @endif
        </div>
```

Gambar 3. 118 HTML Cart bagian change voucher

Pada bagian HTML cart seperti gambar 3.118 juga terdapat perubahan dikarenakan ada kondisi baru. Jika discevent bernilai *null* maka tidak akan menampilkan info-info event. Namun, jika discevent tidak *null* maka akan terisi nama toko, nama item, nama event, dan tanggal event dimulai dan berakhir

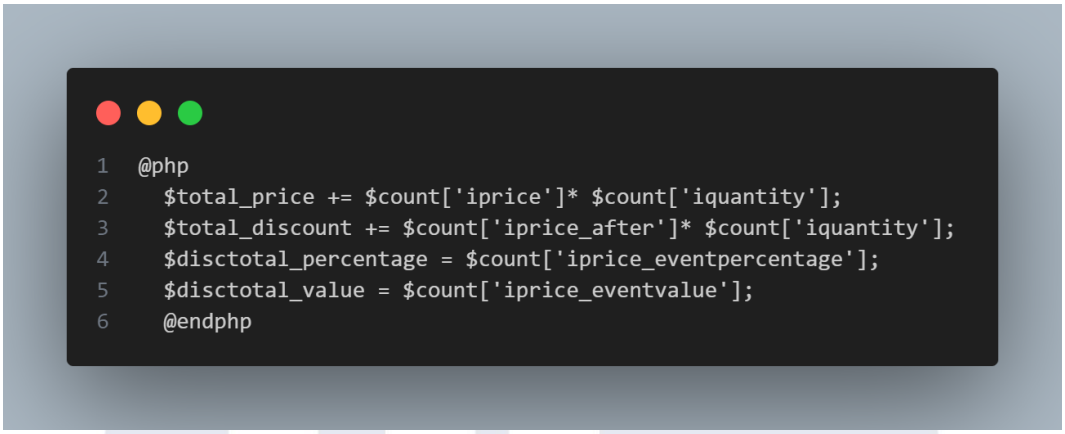


Gambar 3. 119 Tampilan untuk change Event ketika NULL

Pada gambar 3.119 merupakan hasil tampilan ketika discevent bernilai null maka tidak menampilkan info mengenai event.

Gambar 3. 120 Tampilan untuk change Event

Pada gambar 3.120 merupakan hasil tampilan ketika discevent bernilai tidak *null* maka akan terdapat info mengenai nama toko, nama item, nama event, dan tanggal event dimulai dan berakhir.



```php
@php
  $total_price += $count['iprice']* $count['iquantity'];
  $total_discount += $count['iprice_after']* $count['iquantity'];
  $disctotal_percentage = $count['iprice_eventpercentage'];
  $disctotal_value = $count['iprice_eventvalue'];
  @endphp
```

Gambar 3. 121 Perubahan HTML Cart

Pada bagian php di HTML cart juga ada mengalami perubahan yang sebelumnya pada disctotal_percentage ini dikali dengan jumlah quantity tapi untuk sekarang tidak diperlukan seperti pada gambar 3.121.

```
<div class="row">
            <div class="col-md-9">
              <h5 class="card-text fw-bold text-danger">Discount Event (Percentage)</h5>
            </div>
            <div class="col-md-2">
              <p class="text-danger">
                <input name="discount_percentage" type="number" class="form-control" value="{{$disctotal_percentage}}" aria-label="Sizing example input" aria-describedby="inputGroup-sizing-default" hidden />
                Rp. {{number_format($disctotal_percentage)}}
              </p>
            </div>
            <div class="col-md-9">
              <h5 class="card-text fw-bold text-danger">Discount Event (Value)</h5>
            </div>
            <div class="col-md-3">
              <p class="text-danger">
                <input name="discount_value" type="number" class="form-control" value="{{$disctotal_value}}" aria-label="Sizing example input" aria-describedby="inputGroup-sizing-default" hidden />
                Rp. {{number_format($disctotal_value)}}
              </p>
            </div>
```

Gambar 3. 122 Perubahan HTML Cart Discount Event

Selain itu, terdapat perubahan juga pada saat menampilkan *discount* event. Sebelumnya, pada discount event percentage untuk bisa mendapatkan hasil discount percentage perlu mengurangi total price dengan disctotal_ percentage tapi untuk sekarang ini tidak perlu hanya memasukan value dari disctotal_ percentage seperti pada gambar 3.122.

14) Perbaikan Transaksi Detail

a) 4 Oktober – 5 Oktober

Pada kegiatan ini, *developer* melakukan perbaikan pada trasnasksi detail yang sebelumnya data yang diadalam transaksi detail setelah user melakukan pembayaran dan masuk kedalam history dapat berubah-

ubah. Hal ini bisa terjadi dikarenakan pada table transaction_hdr tidak memiliki field yang menampung hasil setelah melakukan transaksi. Oleh karena itu, ada penambahan *field* pada table tersebut. Namun sebelum masuk kedalam *table* transaction _hdr. Data pembelian terlebih dahulu masuk kedalam *table* payment. Dari *table* payment baru diteruskan kedalam *table* transaction_hdr.

```
<div class="row">
            <div class="col-md-9">
              <h5 class="card-text fw-bold text-danger">Discount Event (Percentage)</h5>
            </div>
            <div class="col-md-2">
              <p class="text-danger">
                <input name="discount_percentage" type="number" class="form-control" value="{{$disctotal_percentage}}" aria-label="Sizing example input" aria-describedby="inputGroup-sizing-default" hidden />
                Rp. {{number_format($disctotal_percentage)}}
              </p>
            </div>
            <div class="col-md-9">
              <h5 class="card-text fw-bold text-danger">Discount Event (Value)</h5>
            </div>
            <div class="col-md-3">
              <p class="text-danger">
                <input name="discount_value" type="number" class="form-control" value="{{$disctotal_value}}" aria-label="Sizing example input" aria-describedby="inputGroup-sizing-default" hidden />
                Rp. {{number_format($disctotal_value)}}
              </p>
            </div>
```

Gambar 3. 123 Cart Discount percentage & discount value

Ada beberapa data yang perlu dimasukan kedalam *table* yaitu discount event percentage dan discount event value. Kedua data ini yang nanti akan dimasukan kedalam table agar value dari discount tersebut bisa dijadikan sebagai history transaction. Gambar 3.123 adalah HTML cart untuk discount event pada HTML cart ini ada perubahan yaitu adanya penambahan 1 baris code yaitu input yang bersifat *hidden* dan bertype *number*. Input ini berguna agar data bisa dimasukan kedalam *table*.

```
Payment::create([
            'vsecret_code' => $external_id,
            'vdescription' => $request->notes,
            'ipayment_status' => $response->status,
            'vpayment_link' => $response->invoice_url,
            'iamount' => $request->amount,
            'idiscount_percentage' => $request->discount_percentage,
            'idiscount_value' => $request->discount_value,
            'vcrea' => Auth::user()->email,
            'dcrea' => Carbon::now(),
        ]);
```

Gambar 3. 124 Xendit Controller (Table Payment)

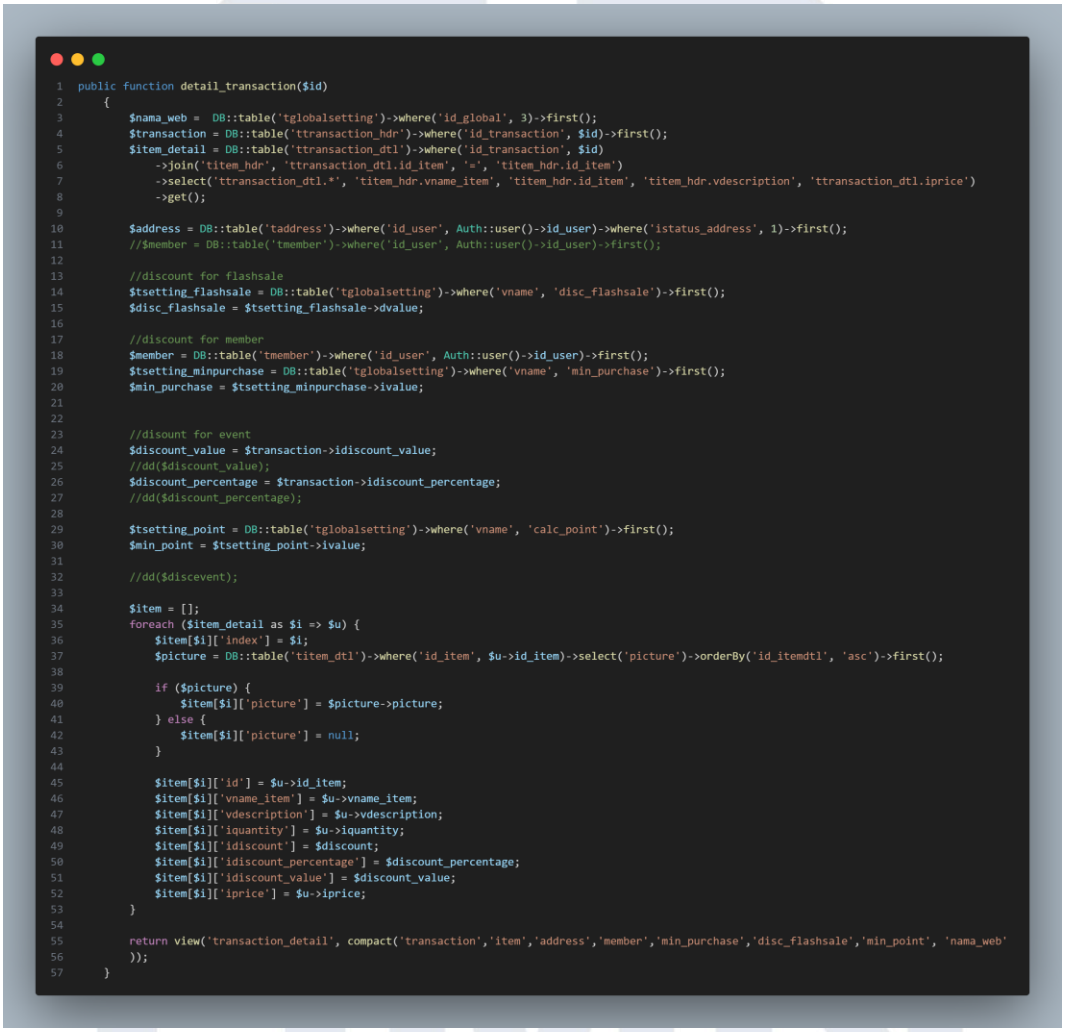
Berikutnya adanya perbaikan di *Xendit controller* karena setelah user melakukan *checkout* maka data tersebut terolah di *Xendit controller*

sebagai *payment*. Perubahan yang terjadi seperti pada gambar 3.124 pada *table* payment juga perlu dimasukkan data dari discount_percentage dan discount value. Cara dimasukan ini dari nama input yang telah ada di HTML Cart. Nama yang ada di HTML Cart harus sama saat memasukkan data ke table payment.

```php
if (is_null($member)) {
    $tsetting7 = DB::table('tglobalsetting')->where('vname', 'calc_point')->first();
    $min_point = $tsetting7->ivalue;
    $total_point = 0;

    DB::table('ttransaction_hdr')->insert([
        'id_user' => Auth::user()->id_user,
        'id_payment' => $check_payment->id_payment,
        'itotal_quantity' => $count_data,
        'itracking_status' => 0,
        'vaddress' => $request->address,
        'vnote' => $request->notes,
        'idiscount_value' => $request->discount_value,
        'idiscount_percentage' => $request->discount_percentage,
        'itotal_price' => $request->amount,
        'itotal_point' => $total_point,
        'vcrea' => Auth::user()->email,
        'dcrea' => Carbon::now(),
    ]);
} elseif (!is_null($member)) {
    if ($total_price >= $min_purchase) {
        if ($member->istatus_member == 1) {
            $tsetting7 = DB::table('tglobalsetting')->where('vname', 'calc_point')->first();
            $min_point = $tsetting7->ivalue;
            $total_point = $total_price / $min_point;
            //dd($total_point);

            DB::table('ttransaction_hdr')->insert([
                'id_user' => Auth::user()->id_user,
                'id_payment' => $check_payment->id_payment,
                'itotal_quantity' => $count_data,
                'itracking_status' => 0,
                'vaddress' => $request->address,
                'vnote' => $request->notes,
                'idiscount_value' => $request->discount_value,
                'idiscount_percentage' => $request->discount_percentage,
                'itotal_price' => $request->amount,
                'itotal_point' => $total_point,
                'vcrea' => Auth::user()->email,
                'dcrea' => Carbon::now(),
            ]);
        } elseif ($member->istatus_member == 0) {
            $tsetting7 = DB::table('tglobalsetting')->where('vname', 'calc_point')->first();
            $min_point = $tsetting7->ivalue;
            $total_point = 0;

            DB::table('ttransaction_hdr')->insert([
                'id_user' => Auth::user()->id_user,
                'id_payment' => $check_payment->id_payment,
                'itotal_quantity' => $count_data,
                'itracking_status' => 0,
                'vaddress' => $request->address,
                'vnote' => $request->notes,
                'idiscount_value' => $request->discount_value,
                'idiscount_percentage' => $request->discount_percentage,
                'itotal_price' => $request->amount,
                'itotal_point' => $total_point,
                'vcrea' => Auth::user()->email,
                'dcrea' => Carbon::now(),
            ]);
        }
    } elseif ($total_price < $min_purchase) {
        $tsetting7 = DB::table('tglobalsetting')->where('vname', 'calc_point')->first();
        $min_point = $tsetting7->ivalue;
        $total_point = 0;

        DB::table('ttransaction_hdr')->insert([
            'id_user' => Auth::user()->id_user,
            'id_payment' => $check_payment->id_payment,
            'itotal_quantity' => $count_data,
            'itracking_status' => 0,
            'vaddress' => $request->address,
            'vnote' => $request->notes,
            'idiscount_value' => $request->discount_value,
            'idiscount_percentage' => $request->discount_percentage,
            'itotal_price' => $request->amount,
            'itotal_point' => $total_point,
            'vcrea' => Auth::user()->email,
            'dcrea' => Carbon::now(),
        ]);
    }
}
```

Gambar 3. 125 Xendit Controller (Table Transaction_hdr)

Setelah berhasil dimasukkan ke dalam *table* payment maka data tersebut juga perlu dimasukkan ke dalam *table* transaction_hdr seperti pada gambar 3.125



```
public function detail_transaction($id)
    {
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        $transaction = DB::table('ttransaction_hdr')->where('id_transaction', $id)->first();
        $item_detail = DB::table('ttransaction_dtl')->where('id_transaction', $id)
            ->join('titem_hdr', 'ttransaction_dtl.id_item', '=', 'titem_hdr.id_item')
            ->select('ttransaction_dtl.*', 'titem_hdr.vname_item', 'titem_hdr.id_item', 'titem_hdr.vdescription', 'ttransaction_dtl.iprice')
            ->get();

        $address = DB::table('taddress')->where('id_user', Auth::user()->id_user)->where('istatus_address', 1)->first();
        //$member = DB::table('tmember')->where('id_user', Auth::user()->id_user)->first();

        //discount for flashsale
        $tsetting_flashsale = DB::table('tglobalsetting')->where('vname', 'disc_flashsale')->first();
        $disc_flashsale = $tsetting_flashsale->dvalue;

        //discount for member
        $member = DB::table('tmember')->where('id_user', Auth::user()->id_user)->first();
        $tsetting_minpurchase = DB::table('tglobalsetting')->where('vname', 'min_purchase')->first();
        $min_purchase = $tsetting_minpurchase->ivalue;


        //disount for event
        $discount_value = $transaction->idiscount_value;
        //dd($discount_value);
        $discount_percentage = $transaction->idiscount_percentage;
        //dd($discount_percentage);

        $tsetting_point = DB::table('tglobalsetting')->where('vname', 'calc_point')->first();
        $min_point = $tsetting_point->ivalue;

        //dd($discevent);

        $item = [];
        foreach ($item_detail as $i => $u) {
            $item[$i]['index'] = $i;
            $picture = DB::table('titem_dtl')->where('id_item', $u->id_item)->select('picture')->orderBy('id_itemdtl', 'asc')->first();

            if ($picture) {
                $item[$i]['picture'] = $picture->picture;
            } else {
                $item[$i]['picture'] = null;
            }

            $item[$i]['id'] = $u->id_item;
            $item[$i]['vname_item'] = $u->vname_item;
            $item[$i]['vdescription'] = $u->vdescription;
            $item[$i]['iquantity'] = $u->iquantity;
            $item[$i]['idiscount'] = $discount;
            $item[$i]['idiscount_percentage'] = $discount_percentage;
            $item[$i]['idiscount_value'] = $discount_value;
            $item[$i]['iprice'] = $u->iprice;
        }

        return view('transaction_detail', compact('transaction','item','address','member','min_purchase','disc_flashsale','min_point', 'nama_web'
        ));
    }
```

Gambar 3. 126 Function detail_transaction

Perubahan juga terjadi pada *function* Detail_transaction di userpagecontroller. *Developer* ada melakukan penambahan 2 variabel yaitu discount_value dan discount_percentage. Kedua variabel ini digunakan untuk menampung value discount value atau discount percentage dari *variable* transaction yang memanggil dari *table* transaction_hdr. Setelah itu juga ada penambahan pada 2 baris saat melakukan *foreach* yaitu baris untuk mengelompokan value dari discount value dan discount percentage kedalam value item.

15) Perbaikan Tampilan Transaction Detail dan Transaction Payment

a) 6 Oktober – 6 Oktober

```php
@php
    $total_quantity = 0;
    $total_price = 0;
    $disctotal_percentage = 0;
    $disctotal_value = 0;
    @endphp
```

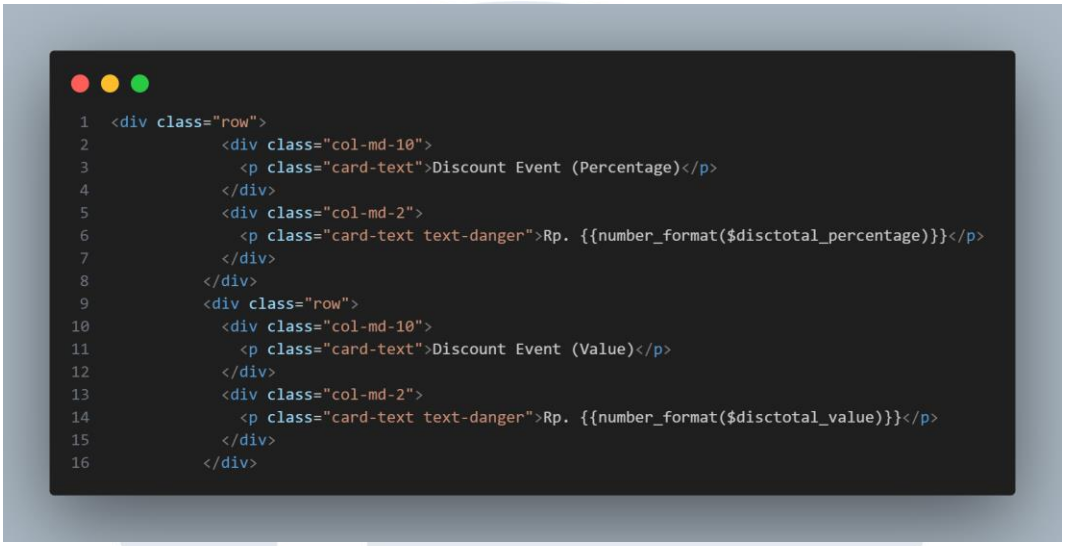
Gambar 3. 127 Perubahan HTML transaction_detail

Setelah melakukan perubahan pada *function* detail_transaction maka pada kegiatan ini, developer melakukan perbaikan pada tampilan transaction_detail. *Developer* pertama-tama melakukan initialisasi variabel disctotal_ percentage dan disctotal_ value seperti pada gambar 3.127.

```php
@php
    $total_price += $count['iprice'];
    $total_quantity += $count['iquantity'];
    $disctotal_percentage = $count['idiscount_percentage'];
    $disctotal_value = $count['idiscount_value'];
    @endphp
```

Gambar 3. 128 Perubahan HTML transaction_detail (I)

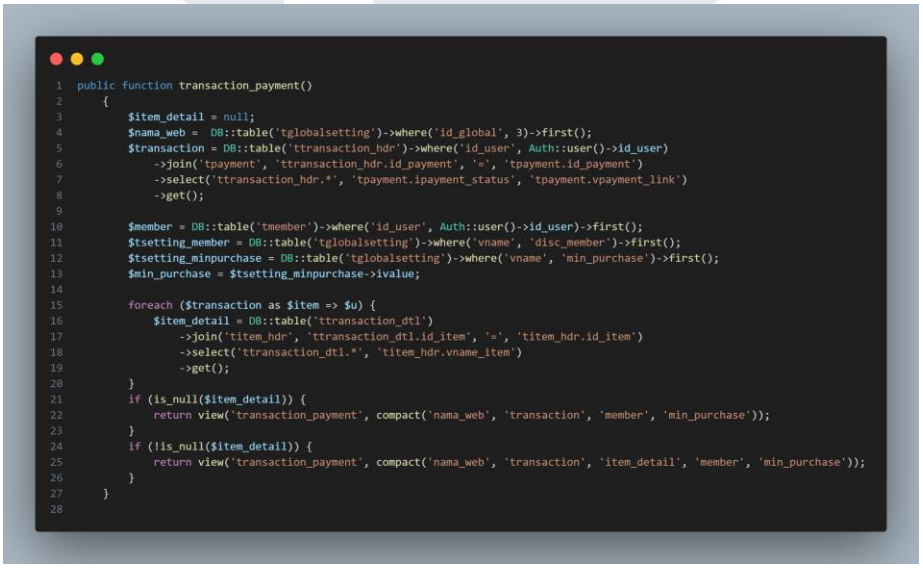
Setelah itu *developer* melakukan pengisian pada variabel tersebut dari *foreach* yang telah dilakukan pada HTML seperti pada gambar 3.128.



```
<div class="row">
            <div class="col-md-10">
              <p class="card-text">Discount Event (Percentage)</p>
            </div>
            <div class="col-md-2">
              <p class="card-text text-danger">Rp. {{number_format($disctotal_percentage)}}</p>
            </div>
          </div>
          <div class="row">
            <div class="col-md-10">
              <p class="card-text">Discount Event (Value)</p>
            </div>
            <div class="col-md-2">
              <p class="card-text text-danger">Rp. {{number_format($disctotal_value)}}</p>
            </div>
          </div>
```

Gambar 3. 129 Perubahan HTML transaction_detail (II)

Pada tampilannya *developer* hanya mengisikan varibale yang telah terisi value discount tersebut seperti pada gambar 3.129.



```
public function transaction_payment()
    {
        $item_detail = null;
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        $transaction = DB::table('ttransaction_hdr')->where('id_user', Auth::user()->id_user)
            ->join('tpayment', 'ttransaction_hdr.id_payment', '=', 'tpayment.id_payment')
            ->select('ttransaction_hdr.*', 'tpayment.ipayment_status', 'tpayment.vpayment_link')
            ->get();

        $member = DB::table('tmember')->where('id_user', Auth::user()->id_user)->first();
        $tsetting_member = DB::table('tglobalsetting')->where('vname', 'disc_member')->first();
        $tsetting_minpurchase = DB::table('tglobalsetting')->where('vname', 'min_purchase')->first();
        $min_purchase = $tsetting_minpurchase->ivalue;

        foreach ($transaction as $item => $u) {
            $item_detail = DB::table('ttransaction_dtl')
                ->join('titem_hdr', 'ttransaction_dtl.id_item', '=', 'titem_hdr.id_item')
                ->select('ttransaction_dtl.*', 'titem_hdr.vname_item')
                ->get();
        }
        if (is_null($item_detail)) {
            return view('transaction_payment', compact('nama_web', 'transaction', 'member', 'min_purchase'));
        }
        if (!is_null($item_detail)) {
            return view('transaction_payment', compact('nama_web', 'transaction', 'item_detail', 'member', 'min_purchase'));
        }
    }
```

Gambar 3. 130 Perbaikan Function Transaction_payment

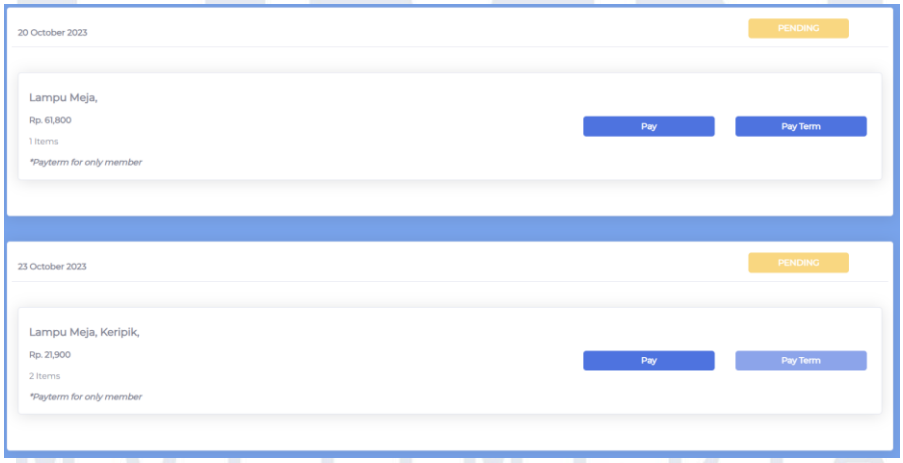
Selain memperbaiki tampilan transaction_detail, *developer* juga melakukan perbaikan pada tampilan transaction_payment untuk payterm. Pertama-tama *developer* melakukan penambahan variable min_purchase yang diambil dari table global setting. Perubahan ini terjadi agar button dari payterm ini juga tidak *active* jika user membeli

dibawah minimal transaksi. Variabel min_purchase dipassing ke tampilan transaction_payment seperti terlihat pada gambar 3.130.

```
@if (is_null($member))
                <a href='#' style="width: inherit;">
                  <button class="btn btn-primary w-100 fw-bold" disabled>Pay Term</button>
                </a>
                @endif
                @if (!is_null($member))
                {{-- Member Active --}}
                @if ($member->istatus_member == 1 && ($u->itotal_price >= $min_purchase))
                <a href="/transaction/payterm/detail/{{$u->id_transaction}}" target="_blank" style="width: inherit;">
                  <button class="btn btn-primary w-100 fw-bold">Pay Term</button>
                </a>
                @endif

                @if ($member->istatus_member == 1 && ($u->itotal_price < $min_purchase))
                <a href="#"  style="width: inherit;">
                  <button class="btn btn-primary w-100 fw-bold" disabled>Pay Term</button>
                </a>
                @endif

                {{-- Member Inactive --}}
                @if ($member->istatus_member == 0)
                <a href='#' style="width: inherit;">
                  <button class="btn btn-primary w-100 fw-bold" disabled>Pay Term</button>
                </a>
                @endif
                @endif
```

Gambar 3. 131 Perubahan Tampilan Transaction_payment

Pada tampilan transaction_payment seperti gambar 3.131 juga terjadi perubahan kondisi untuk member yang memiliki status bernilai 1 atau *active*. Perubahan kondisi ini mengikuti dengan minimal pembelian. Developer menambahkan kondisi jika total price lebih besar dari min_purchase maka button payterm akan berfungsi jika total price kurang dari min_purchase maka button payterm akan disable



Gambar 3. 132 Tampilan Button Payterm

Pada gambar 3. 132 merupakan hasil dari kondisi yang telah dibuat pada HTML transaction_payment yang terdapat button disable dan button enable.

16) Pemeriksaan Validasi dari Project

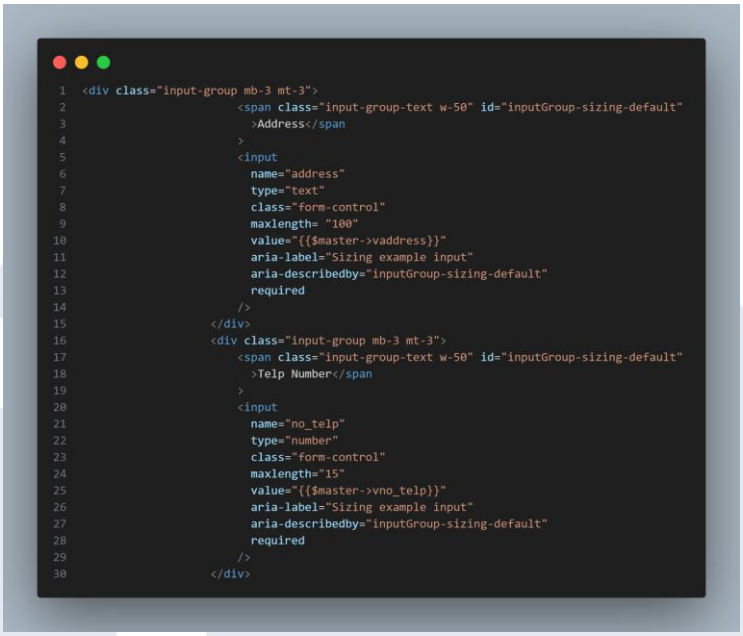
a) 7 Oktober – 10 Oktober

Pada kegiatan ini, *developer* melakukan pengcekan validasi dari tugas-tugas yang telah ada. Pengcekan ini seperti mengecek validasi apakah sama max value dari tuasg-tugas yang ada sama dengan max value di database. Oleh karena itu, *developer* melakukan pengecekan.

```
<div class="row mt-3 justify-content-center">
      <div class="col-md-10">
        <div class="mb-3">
          <label for="exampleFormControlTextarea1" class="form-label fw-bold">Notes</label>
          <textarea name="notes" class="form-control" id="exampleFormControlTextarea1" rows="2" maxlength="255"></textarea>
        </div>
      </div>
    </div>
```

Gambar 3. 133 Cart Blade

Pertama-tama *developer* melakukan pengecekan validasi pada tampilan cart. Untuk tampilan cart developer melakukan pengecekan pada baris note apakah note hanya bisa input sebanyak 255 character. Developer melakukan penambahan *max length* seperti pada gambar 3.133 dikarenakan cart karena sebelum melakukan penambahan bisa menginput lebih dari 255 character dan akan terjadi *error* ketika users melakukan *checkout* maka dari itu ada *max length* sebagai pembatas inputan.



```
<div class="input-group mb-3 mt-3">
                        <span class="input-group-text w-50" id="inputGroup-sizing-default"
                          >Address</span
                        >
                        <input
                          name="address"
                          type="text"
                          class="form-control"
                          maxlength= "100"
                          value="{{$master->vaddress}}"
                          aria-label="Sizing example input"
                          aria-describedby="inputGroup-sizing-default"
                          required
                        />
                    </div>
                    <div class="input-group mb-3 mt-3">
                        <span class="input-group-text w-50" id="inputGroup-sizing-default"
                          >Telp Number</span
                        >
                        <input
                          name="no_telp"
                          type="number"
                          class="form-control"
                          maxlength="15"
                          value="{{$master->vno_telp}}"
                          aria-label="Sizing example input"
                          aria-describedby="inputGroup-sizing-default"
                          required
                        />
                    </div>
```

Gambar 3. 134 Mastertoko_edit

Selain itu developer juga menambahkan validasi berupa *max length* pada tampilan mastertoko_edit pada bagian address dan nomor telepon seperti pada gambar 3.134.

```
<div class="input-group mb-3 mt-3">
                        <span class="input-group-text w-50" id="inputGroup-sizing-default"
                          >Item Name</span
                        >
                        <input
                          name="name"
                          type="text"
                          class="form-control"
                          value="{{$master->vname_item}}"
                          maxlength="100"
                          aria-label="Sizing example input"
                          aria-describedby="inputGroup-sizing-default"
                          required
                        />
                    </div>
```

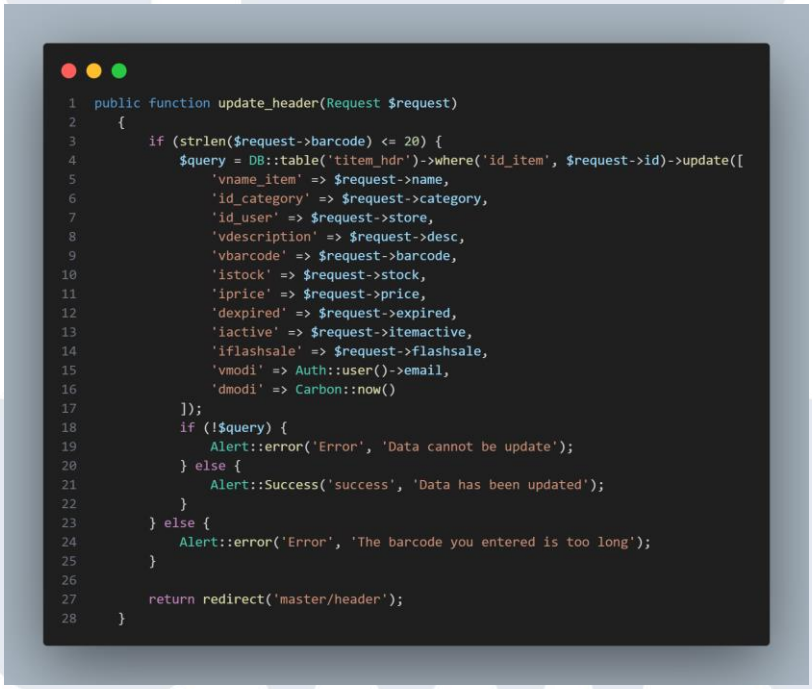
Gambar 3. 135 Masterheader_edit (I)

Hal dilakukan juga sama pada tampilan master header_edit dengan menambahkan *max length* pada bagian item name, dan description seperti pada gambar 3.135.

```
<div class="input-group mb-3 mt-3">
                        <span class="input-group-text w-50" id="inputGroup-sizing-default"
                          >Description</span
                        >
                        <textarea
                          name="desc"
                          type="text"
                          class="form-control"
                          aria-label="Sizing example input"
                          maxlength="255"
                          aria-describedby="inputGroup-sizing-default"required
                        >{{$master->vdescription}}</textarea>
                    </div>
```

Gambar 3. 136 Masterheader_edit (II)

Hal dilakukan juga sama pada tampilan master header_edit dengan menambahkan *max length* pada bagian item name, dan description seperti pada gambar 3.136.



```
public function update_header(Request $request)
    {
        if (strlen($request->barcode) <= 20) {
            $query = DB::table('titem_hdr')->where('id_item', $request->id)->update([
                'vname_item' => $request->name,
                'id_category' => $request->category,
                'id_user' => $request->store,
                'vdescription' => $request->desc,
                'vbarcode' => $request->barcode,
                'istock' => $request->stock,
                'iprice' => $request->price,
                'dexpired' => $request->expired,
                'iactive' => $request->itemactive,
                'iflashsale' => $request->flashsale,
                'vmodi' => Auth::user()->email,
                'dmodi' => Carbon::now()
            ]);
            if (!$query) {
                Alert::error('Error', 'Data cannot be update');
            } else {
                Alert::Success('success', 'Data has been updated');
            }
        } else {
            Alert::error('Error', 'The barcode you entered is too long');
        }

        return redirect('master/header');
    }
```

Gambar 3. 137 Function Update_header

Selain ada pada tampilan, *Developer* juga melakukan validasi dengan perubahan *function* pada update_header. Developer melakukan perubahan dengan kondisi IF untuk membatasi barcode tidak lebih dari 20 character jika barcode lebih dari 20 character maka tidak bisa terupdate dan muncul notifikasi *alert* error seperti pada gambar 3.137.

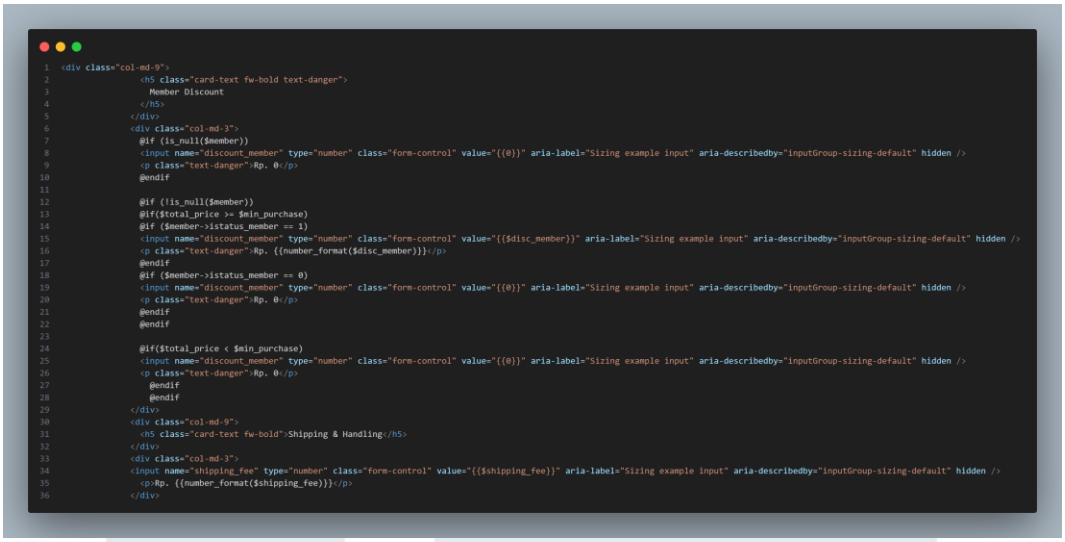
3.2.2.3 Pengembangan Return Item Lanjutan (Admin)

1) **Pengembangan Pengurangan Point, Penambahan Item dan Penambahan Saldo**

   a) **11 Oktober – 18 Oktober**

Pada kegiatan ini, *developer* mendapatkan tugas untuk mengurangi point, penambahan stock item, dan penambahan saldo. Penambahan ini terjadi ketika user yang menjadi admin toko saat mengklik *accept* maka akan secara otomatis point akan berkurang, stock item akan bertambah dan ada penambahan pada saldo.

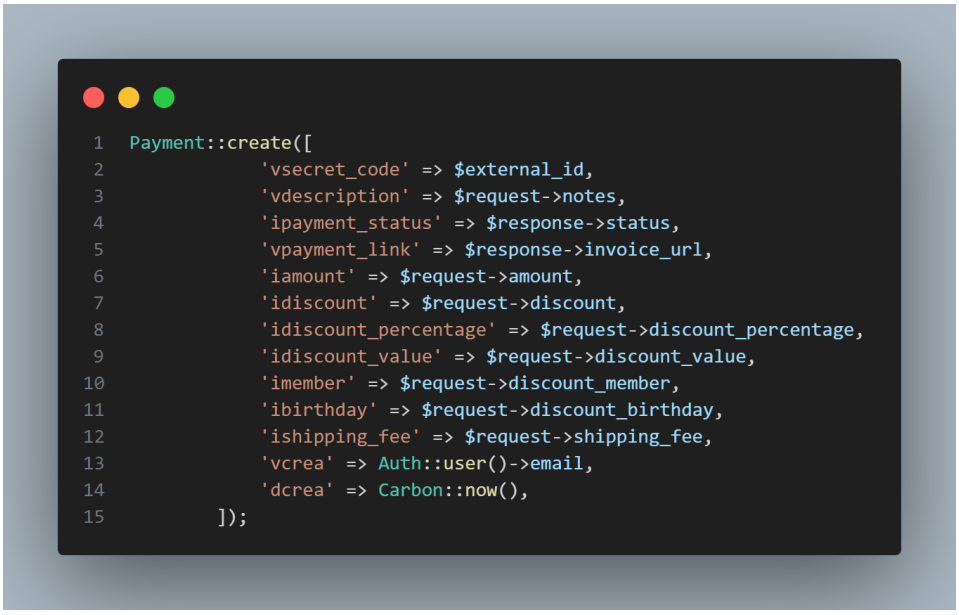
Pertama-tama terdapat penambahan kolom pada *table* payment, transaction_hdr, dan transaction_dtl yang sebelumnya pada *table* payment dan *table* transaction_hdr telah ada penambahan kolom discount value dan discount percentage maka ditambah lagi kolom untuk discount, discount member, discount birthday, dan shipping fee. Sementara untuk *table* transaction_dtl penambahan kolom seperti *table* transaction_hdr yaitu discount value, discount percentage, discount, discount member, discount birthday, dan shipping fee. Penambhaan kolom ini agar semua data setelah *checkout* bisa tertampung kedalam *database* dan tidak akan dipengaruhi dengan data dari *table* yang lain dan ini untuk memudahkan dalam pengembalian point dan penambahan saldo.



```
<div class="col-md-9">
                <h5 class="card-text fw-bold text-danger">
                  Member Discount
                </h5>
              </div>
              <div class="col-md-3">
                @if (is_null($member))
                <input name="discount_member" type="number" class="form-control" value="{{0}}" aria-label="Sizing example input" aria-describedby="inputGroup-sizing-default" hidden />
                <p class="text-danger">Rp. 0</p>
                @endif

                @if (!is_null($member))
                @if($total_price >= $min_purchase)
                @if ($member->istatus_member == 1)
                <input name="discount_member" type="number" class="form-control" value="{{$disc_member}}" aria-label="Sizing example input" aria-describedby="inputGroup-sizing-default" hidden />
                <p class="text-danger">Rp. {{number_format($disc_member)}}</p>
                @endif
                @if ($member->istatus_member == 0)
                <input name="discount_member" type="number" class="form-control" value="{{0}}" aria-label="Sizing example input" aria-describedby="inputGroup-sizing-default" hidden />
                <p class="text-danger">Rp. 0</p>
                @endif
                @endif

                @if($total_price < $min_purchase)
                <input name="discount_member" type="number" class="form-control" value="{{0}}" aria-label="Sizing example input" aria-describedby="inputGroup-sizing-default" hidden />
                <p class="text-danger">Rp. 0</p>
                  @endif
                  @endif
              </div>
              <div class="col-md-9">
                <h5 class="card-text fw-bold">Shipping & Handling</h5>
              </div>
              <div class="col-md-3">
              <input name="shipping_fee" type="number" class="form-control" value="{{$shipping_fee}}" aria-label="Sizing example input" aria-describedby="inputGroup-sizing-default" hidden />
                <p>Rp. {{number_format($shipping_fee)}}</p>
              </div>
```

Gambar 3. 138 Potongan Code untuk menampung value pada Cart

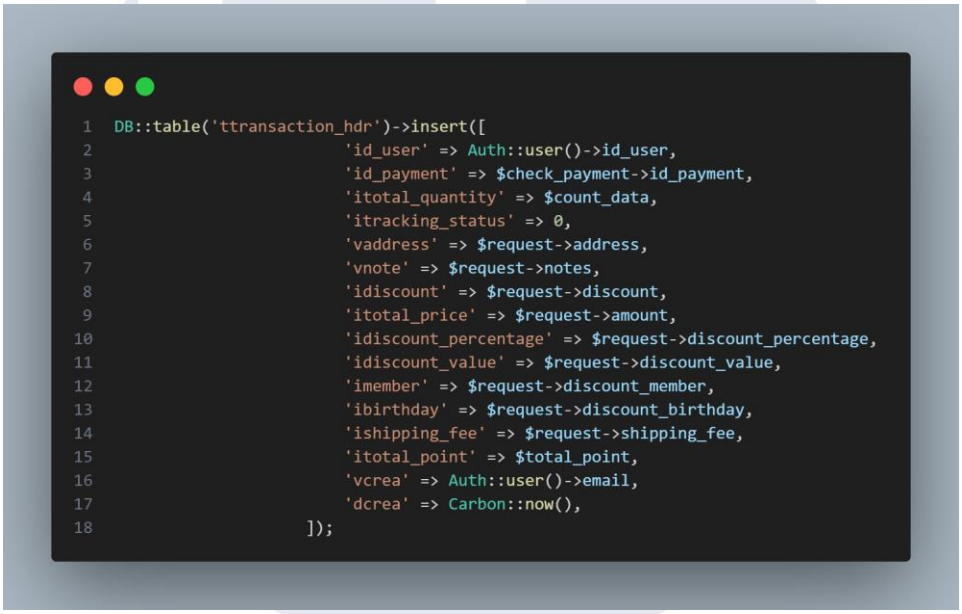
Oleh karena itu, *developer* melakukan perubahan pada tampilan cart seperti perubahan pada transaction detail segala hasil dari value discount hingga discount member maka terdapat penambahan baris input agar baris ini bisa menampung semua value pada suser ingin *checkout* seperti pada gambar 3.138 merupakan potongan code dari discount member dan shipping fee.



```
Payment::create([
            'vsecret_code' => $external_id,
            'vdescription' => $request->notes,
            'ipayment_status' => $response->status,
            'vpayment_link' => $response->invoice_url,
            'iamount' => $request->amount,
            'idiscount' => $request->discount,
            'idiscount_percentage' => $request->discount_percentage,
            'idiscount_value' => $request->discount_value,
            'imember' => $request->discount_member,
            'ibirthday' => $request->discount_birthday,
            'ishipping_fee' => $request->shipping_fee,
            'vcrea' => Auth::user()->email,
            'dcrea' => Carbon::now(),
        ]);
```

Gambar 3. 139 Perubahan table payment

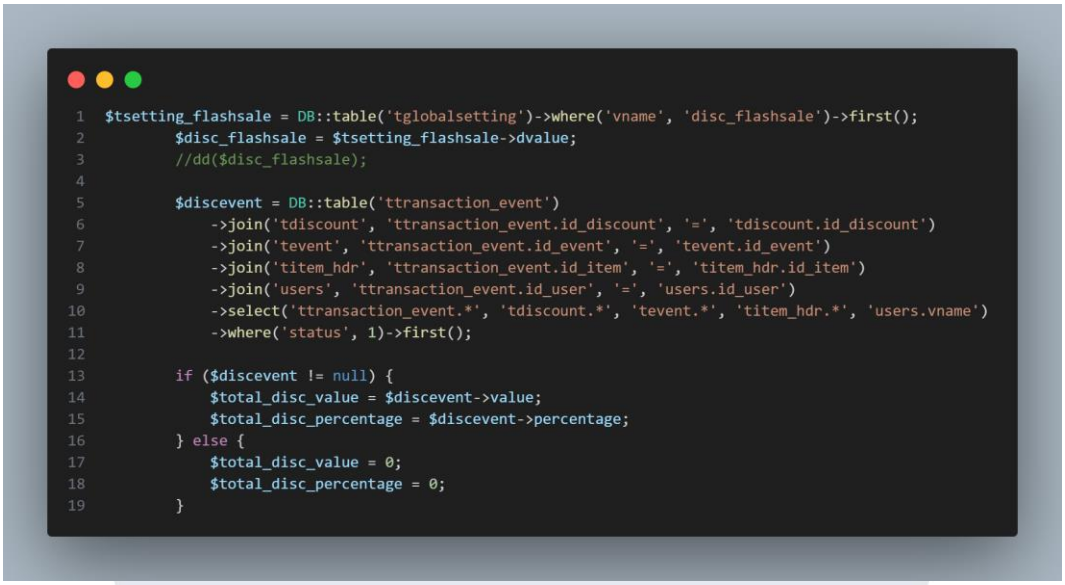
Setelah itu juga ada perubahan pada *xendit controller* yang terjadi pada *table* payment. Perubahan ini diperlukan agar *table* payment dapat menampung value dari hasil *checkout* user ke dalam *table* seperti pada gambar 3.139.



```
DB::table('ttransaction_hdr')->insert([
                        'id_user' => Auth::user()->id_user,
                        'id_payment' => $check_payment->id_payment,
                        'itotal_quantity' => $count_data,
                        'itracking_status' => 0,
                        'vaddress' => $request->address,
                        'vnote' => $request->notes,
                        'idiscount' => $request->discount,
                        'itotal_price' => $request->amount,
                        'idiscount_percentage' => $request->discount_percentage,
                        'idiscount_value' => $request->discount_value,
                        'imember' => $request->discount_member,
                        'ibirthday' => $request->discount_birthday,
                        'ishipping_fee' => $request->shipping_fee,
                        'itotal_point' => $total_point,
                        'vcrea' => Auth::user()->email,
                        'dcrea' => Carbon::now(),
                    ]);
```

Gambar 3. 140 Perubahan table transaction_hdr

Perubahan tidak terjadi pada *table* payment saja tapi juga perubahan pada *table* transaction_hdr seperti pada gambar 3.140. Perubahan tersebut terdapat penambahan mengenai penyimpanan data pada table transaction_hdr



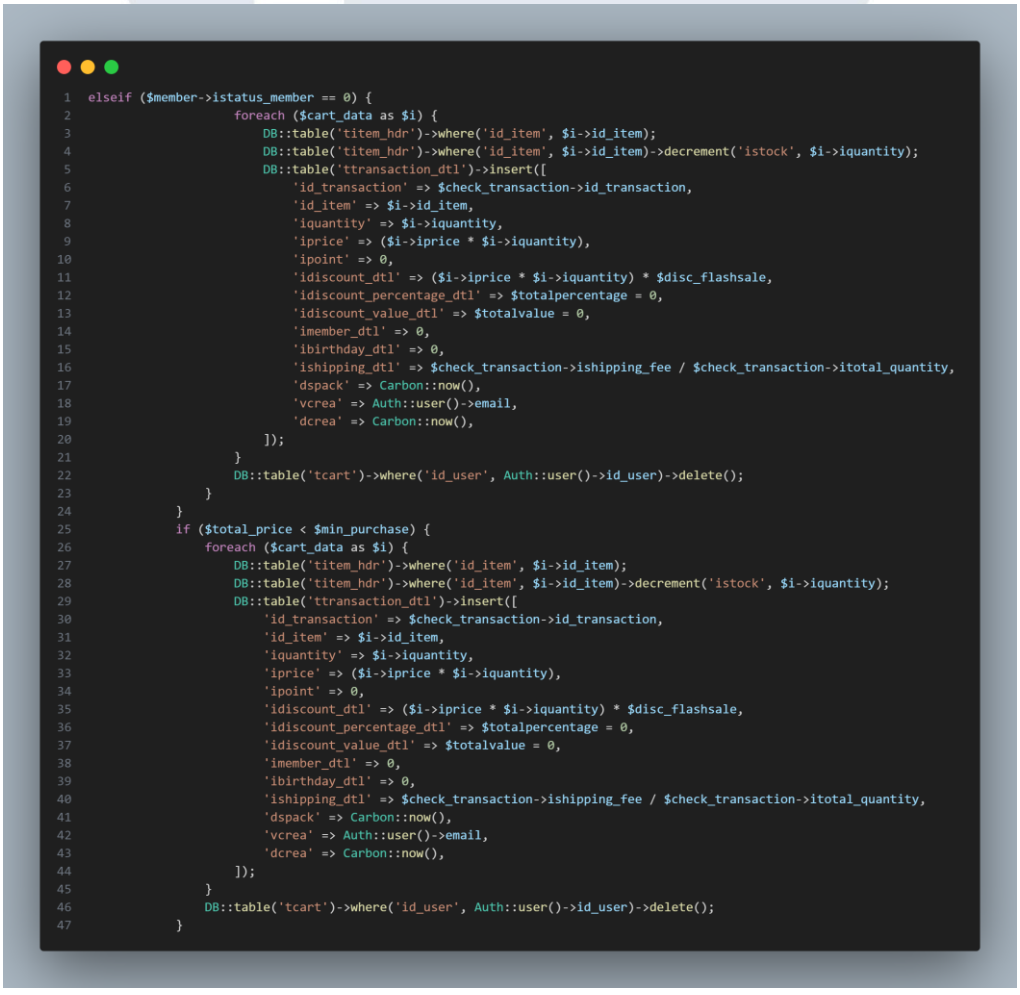
```php
$tsetting_flashsale = DB::table('tglobalsetting')->where('vname', 'disc_flashsale')->first();
        $disc_flashsale = $tsetting_flashsale->dvalue;
        //dd($disc_flashsale);

        $discevent = DB::table('ttransaction_event')
            ->join('tdiscount', 'ttransaction_event.id_discount', '=', 'tdiscount.id_discount')
            ->join('tevent', 'ttransaction_event.id_event', '=', 'tevent.id_event')
            ->join('titem_hdr', 'ttransaction_event.id_item', '=', 'titem_hdr.id_item')
            ->join('users', 'ttransaction_event.id_user', '=', 'users.id_user')
            ->select('ttransaction_event.*', 'tdiscount.*', 'tevent.*', 'titem_hdr.*', 'users.vname')
            ->where('status', 1)->first();

        if ($discevent != null) {
            $total_disc_value = $discevent->value;
            $total_disc_percentage = $discevent->percentage;
        } else {
            $total_disc_value = 0;
            $total_disc_percentage = 0;
        }
```

Gambar 3. 141 Penambhan code pada xendit controller

Berikutnya *developer* juga menambahkan *variable* disc_flashsale yang menampung value dari data di global setting. Tidak hanya itu saja *developer* juga memasukan *variable* discevent pada *xendit controller* agar bisa menampung value discount event sesuai item pada transaction_dtl. Code ini seperti code yang berada di function cart dengan ada 2 kondisi jika discevent tidak bernilai null maka terdapat variable yang menampung value dari discevent baik value berupa percentage atau tidak seperti pada gamabr 3.141.

```php
if (!is_null($check_transaction)) {
        //Non Member
        if (is_null($member)) {
            foreach ($cart_data as $i) {
                DB::table('titem_hdr')->where('id_item', $i->id_item);
                DB::table('titem_hdr')->where('id_item', $i->id_item)->decrement('istock', $i->iquantity);
                DB::table('ttransaction_dtl')->insert([
                    'id_transaction' => $check_transaction->id_transaction,
                    'id_item' => $i->id_item,
                    'iquantity' => $i->iquantity,
                    'iprice' => ($i->iprice * $i->iquantity),
                    'ipoint' => 0,
                    'idiscount_dtl' => ($i->iprice * $i->iquantity) * $disc_flashsale,
                    'idiscount_percentage_dtl' => $totalpercentage = 0,
                    'idiscount_value_dtl' => $totalvalue = 0,
                    'imember_dtl' => 0,
                    'ibirthday_dtl' => 0,
                    'ishipping_dtl' => $check_transaction->ishipping_fee / $check_transaction->itotal_quantity,
                    'dspack' => Carbon::now(),
                    'vcrea' => Auth::user()->email,
                    'dcrea' => Carbon::now(),
                ]);
            }
            DB::table('tcart')->where('id_user', Auth::user()->id_user)->delete();
        }
```

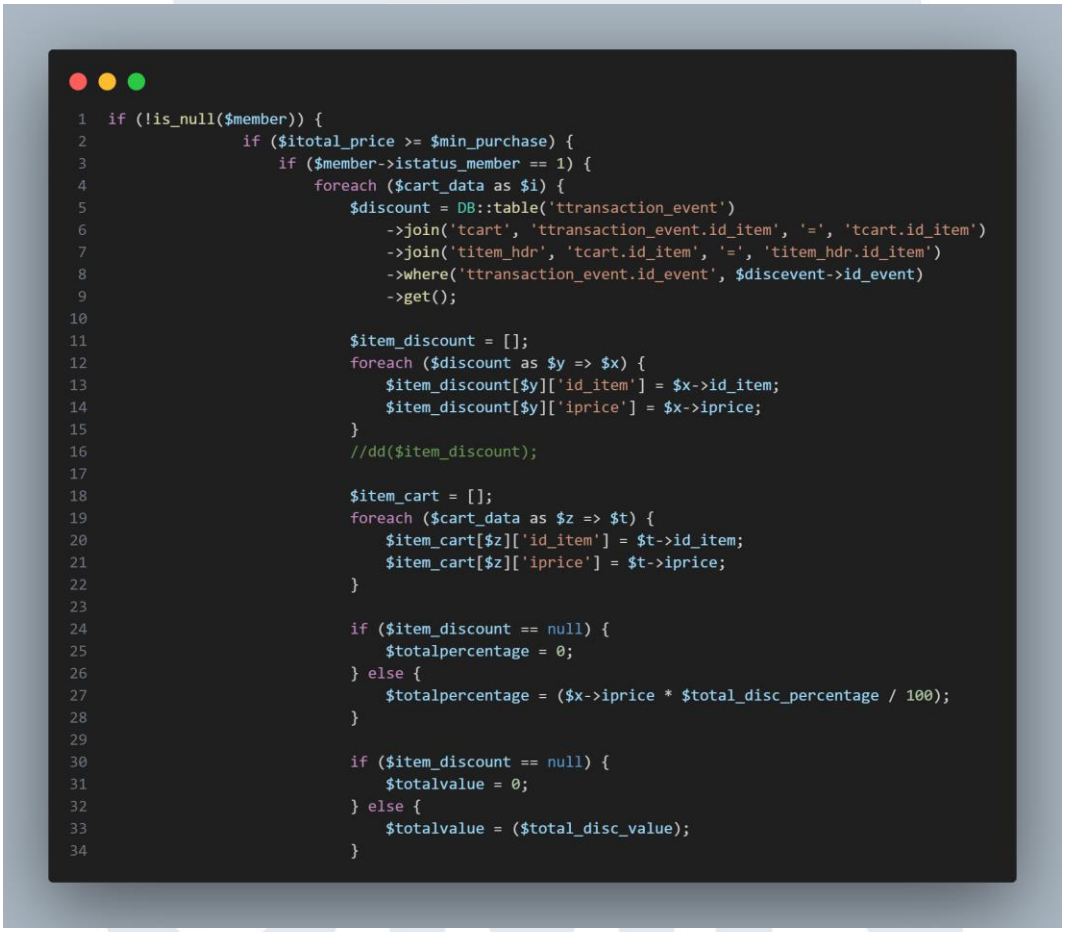
Gambar 3. 142 Insert data Transaction_dtl ketika discount bernilai 0



```php
elseif ($member->istatus_member == 0) {
                    foreach ($cart_data as $i) {
                        DB::table('titem_hdr')->where('id_item', $i->id_item);
                        DB::table('titem_hdr')->where('id_item', $i->id_item)->decrement('istock', $i->iquantity);
                        DB::table('ttransaction_dtl')->insert([
                            'id_transaction' => $check_transaction->id_transaction,
                            'id_item' => $i->id_item,
                            'iquantity' => $i->iquantity,
                            'iprice' => ($i->iprice * $i->iquantity),
                            'ipoint' => 0,
                            'idiscount_dtl' => ($i->iprice * $i->iquantity) * $disc_flashsale,
                            'idiscount_percentage_dtl' => $totalpercentage = 0,
                            'idiscount_value_dtl' => $totalvalue = 0,
                            'imember_dtl' => 0,
                            'ibirthday_dtl' => 0,
                            'ishipping_dtl' => $check_transaction->ishipping_fee / $check_transaction->itotal_quantity,
                            'dspack' => Carbon::now(),
                            'vcrea' => Auth::user()->email,
                            'dcrea' => Carbon::now(),
                        ]);
                    }
                    DB::table('tcart')->where('id_user', Auth::user()->id_user)->delete();
                }
            }
            if ($total_price < $min_purchase) {
                foreach ($cart_data as $i) {
                    DB::table('titem_hdr')->where('id_item', $i->id_item);
                    DB::table('titem_hdr')->where('id_item', $i->id_item)->decrement('istock', $i->iquantity);
                    DB::table('ttransaction_dtl')->insert([
                        'id_transaction' => $check_transaction->id_transaction,
                        'id_item' => $i->id_item,
                        'iquantity' => $i->iquantity,
                        'iprice' => ($i->iprice * $i->iquantity),
                        'ipoint' => 0,
                        'idiscount_dtl' => ($i->iprice * $i->iquantity) * $disc_flashsale,
                        'idiscount_percentage_dtl' => $totalpercentage = 0,
                        'idiscount_value_dtl' => $totalvalue = 0,
                        'imember_dtl' => 0,
                        'ibirthday_dtl' => 0,
                        'ishipping_dtl' => $check_transaction->ishipping_fee / $check_transaction->itotal_quantity,
                        'dspack' => Carbon::now(),
                        'vcrea' => Auth::user()->email,
                        'dcrea' => Carbon::now(),
                    ]);
                }
                DB::table('tcart')->where('id_user', Auth::user()->id_user)->delete();
            }
```

Gambar 3. 143 Insert data Transaction_dtl ketika discount bernilai 0 (I)

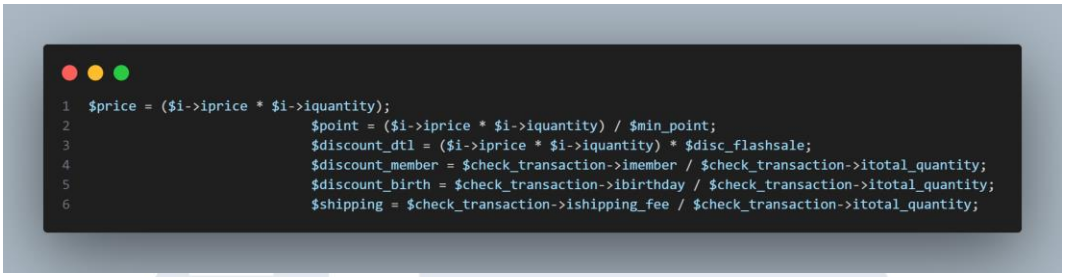
Lalu pada saat pengsisin data ke table transaction_dtl maka terdapat kondisi IF jika *variable* member bernilai *null* maka semua discount akan bernilai 0 kecuali discount dari flashsale. Begitu juga jika member memiliki status bernilai 0 dan total price kurang dari minimal purchase maka semua discount akan bernilai 0 dapat terlihat seperti gambar 3.142 dan 3.143



```
if (!is_null($member)) {
                if ($itotal_price >= $min_purchase) {
                    if ($member->istatus_member == 1) {
                        foreach ($cart_data as $i) {
                            $discount = DB::table('ttransaction_event')
                                ->join('tcart', 'ttransaction_event.id_item', '=', 'tcart.id_item')
                                ->join('titem_hdr', 'tcart.id_item', '=', 'titem_hdr.id_item')
                                ->where('ttransaction_event.id_event', $discevent->id_event)
                                ->get();

                            $item_discount = [];
                            foreach ($discount as $y => $x) {
                                $item_discount[$y]['id_item'] = $x->id_item;
                                $item_discount[$y]['iprice'] = $x->iprice;
                            }
                            //dd($item_discount);

                            $item_cart = [];
                            foreach ($cart_data as $z => $t) {
                                $item_cart[$z]['id_item'] = $t->id_item;
                                $item_cart[$z]['iprice'] = $t->iprice;
                            }

                            if ($item_discount == null) {
                                $totalpercentage = 0;
                            } else {
                                $totalpercentage = ($x->iprice * $total_disc_percentage / 100);
                            }

                            if ($item_discount == null) {
                                $totalvalue = 0;
                            } else {
                                $totalvalue = ($total_disc_value);
                            }
```

Gambar 3. 144 Penambahan code untuk item yang ter-discount

Hal ini berbanding balik dengan yang sebelumnya karena ketika variable member tidak bernilai *null* dan status member bernilai 1 maka value discount akan terisi sesuai hasil *checkout* user. Pada status member bernilai 1 maka developer menambahkan code yang dari *function* cart untuk mengetahui barang yang mendapatkan discount dari discevent seperti pada gambar 3.144.



```
$price = ($i->iprice * $i->iquantity);
                        $point = ($i->iprice * $i->iquantity) / $min_point;
                        $discount_dtl = ($i->iprice * $i->iquantity) * $disc_flashsale;
                        $discount_member = $check_transaction->imember / $check_transaction->itotal_quantity;
                        $discount_birth = $check_transaction->ibirthday / $check_transaction->itotal_quantity;
                        $shipping = $check_transaction->ishipping_fee / $check_transaction->itotal_quantity;
```

Gambar 3. 145 Penambahan variable penampung untuk perhitungan discount

Setelah itu developer membuat beberapa variable dari hasil perhitungan yang ingin dimasukkan ke table transaction_dtl seperti pada gambar 3.145 serta ada beberapa variabel seperti discount member, discount bithday dan shiping fee yang valuenya dibagi sesuai total quantity per item.

```
DB::table('titem_hdr')->where('id_item', $i->id_item)->first();
                        DB::table('titem_hdr')->where('id_item', $i->id_item)->decrement('istock', $i->iquantity);
                        if ($item_discount != null && $i->id_item == $x->id_item) {
                            DB::table('ttransaction_dtl')->insert([
                                'id_transaction' => $check_transaction->id_transaction,
                                'id_item' => $i->id_item,
                                'iquantity' => $i->iquantity,
                                'iprice' => $price,
                                'ipoint' => $point,
                                'idiscount_dtl' => $discount_dtl,
                                'idiscount_percentage_dtl' => $totalpercentage,
                                'idiscount_value_dtl' => $totalvalue,
                                'imember_dtl' => $discount_member,
                                'ibirthday_dtl' => $discount_birth,
                                'ishipping_dtl' => $shipping,
                                'dspack' => Carbon::now(),
                                'vcrea' => Auth::user()->email,
                                'dcrea' => Carbon::now(),
                            ]);

```

Gambar 3. 146 Insert data ke transaction_dtl (I)

Setelah itu, ketika melakukan *insert* data ke *table* transaction_dtl terdapat kondisi IF jika variabel item_discount tidak bernilai null dan id_item dari barang tersebut sama dengan id_item dari varibel discount maka value dari discount percentage dan discunt value akan terisi sesuai diskon seperti pada gambar 3.146.

```
} else {
                        DB::table('ttransaction_dtl')->insert([
                            'id_transaction' => $check_transaction->id_transaction,
                            'id_item' => $i->id_item,
                            'iquantity' => $i->iquantity,
                            'iprice' => $price,
                            'ipoint' => $point,
                            'idiscount_dtl' => $discount_dtl,
                            'idiscount_percentage_dtl' => $totalpercentage = 0,
                            'idiscount_value_dtl' => $totalvalue = 0,
                            'imember_dtl' => $discount_member,
                            'ibirthday_dtl' => $discount_birth,
                            'ishipping_dtl' => $shipping,
                            'dspack' => Carbon::now(),
                            'vcrea' => Auth::user()->email,
                            'dcrea' => Carbon::now(),
                        ]);
```

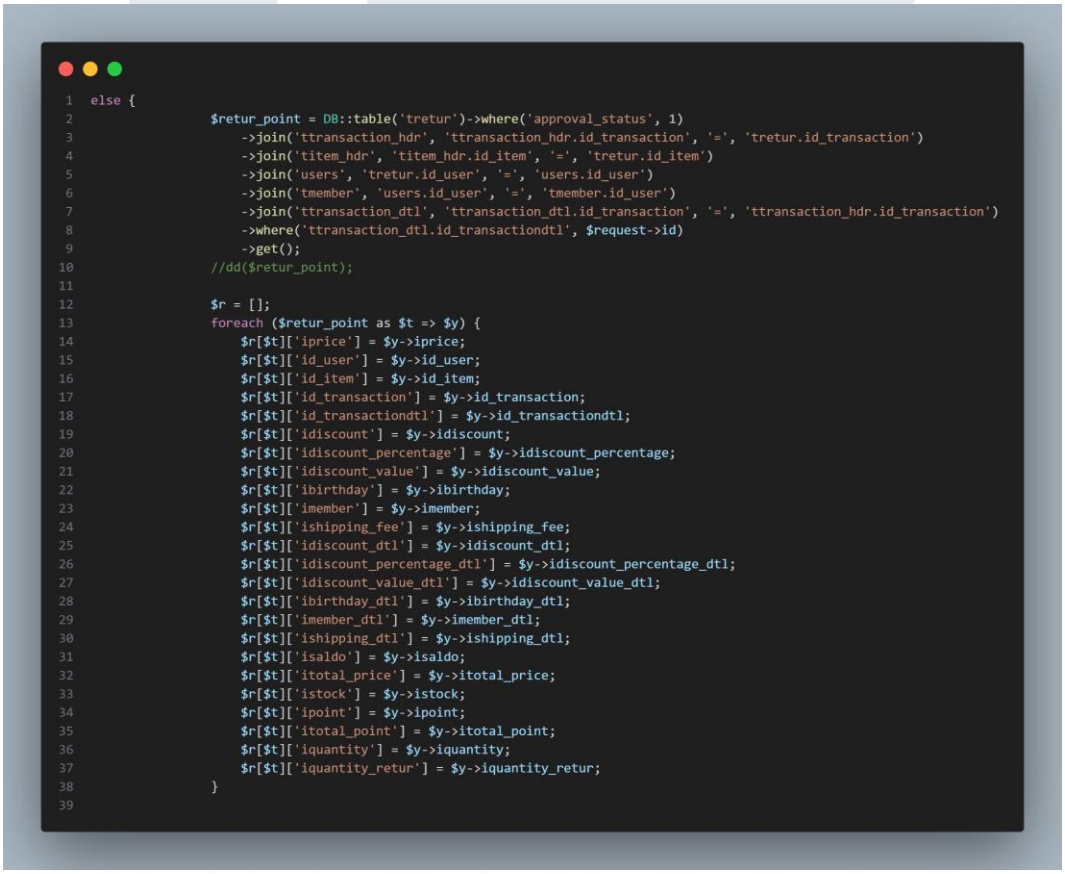
Gambar 3. 147 Insert data ke transaction_dtl (II)

Jika kondisi tersebut tidak terpenuhi maka value dari discount percentage dan discunt value bernilai 0 seperti pada gambar 3.147. Hal ini perlu dilakukan agar ketika melakukan *foreach* data yang dimasukkan ke *table* trasnaction_dtl tersebut akan menyesuaikan item dengan discountnya

```
public function reason_return_admin_accept(Request $request)
    {
        $retur_data =  DB::table('tretur')->where('id_transactiondtl', $request->id)->get();

        $data = [];
        foreach ($retur_data as $e => $l) {
            $data[$e]['id_retur'] = $l->id_retur;
            $data[$e]['approval_status'] = $l->approval_status;
        }

        if ($l->approval_status == 2) {
            $retur = DB::table('tretur')->where('id_retur', $l->id_retur)
                ->update([
                    'approval_status' => 2,
                ]);
            Alert::error('Error', 'Data cannot change to approval');
        } else {
            $retur = DB::table('tretur')->where('id_transactiondtl', $request->id)
                ->update([
                    'approval_status' => 1,
                ]);

            //dd($retur);
            if (!$retur) {
                Alert::error('Error', 'Data cannot approve again');
            } else {
```

Gambar 3. 148 Master Controller Accept return

Developer juga melanjutkan pada *function* reason_return_admin_accept pada master controller. *Developer* menambahkan *variable* retur_data yang memanggil dari *table* tretur dengan kondisi where dari id_trasnactiondtl sesuai *request* dari user yang menjadi admin lalu developer melakukan *foreach* untuk menaggil kolom sesuai kebutuhan user yaitu id_retur dan approval status. Hal ini dibutuhkan untuk membatasi admin untuk *accept* berkali-kali dan setelah reject tidak bisa accept maka terdapat kondisi jika approval status bernilai 2 maka akan tetap bernilai 2 dan muncul notifikasi *error* jika tidak maka approval status akan berubah menjadi 1 dapat terlihat pada gambar 3.148.



```
else {
            $retur_point = DB::table('tretur')->where('approval_status', 1)
                ->join('ttransaction_hdr', 'ttransaction_hdr.id_transaction', '=', 'tretur.id_transaction')
                ->join('titem_hdr', 'titem_hdr.id_item', '=', 'tretur.id_item')
                ->join('users', 'tretur.id_user', '=', 'users.id_user')
                ->join('tmember', 'users.id_user', '=', 'tmember.id_user')
                ->join('ttransaction_dtl', 'ttransaction_dtl.id_transaction', '=', 'ttransaction_hdr.id_transaction')
                ->where('ttransaction_dtl.id_transactiondtl', $request->id)
                ->get();
            //dd($retur_point);

            $r = [];
            foreach ($retur_point as $t => $y) {
                $r[$t]['iprice'] = $y->iprice;
                $r[$t]['id_user'] = $y->id_user;
                $r[$t]['id_item'] = $y->id_item;
                $r[$t]['id_transaction'] = $y->id_transaction;
                $r[$t]['id_transactiondtl'] = $y->id_transactiondtl;
                $r[$t]['idiscount'] = $y->idiscount;
                $r[$t]['idiscount_percentage'] = $y->idiscount_percentage;
                $r[$t]['idiscount_value'] = $y->idiscount_value;
                $r[$t]['ibirthday'] = $y->ibirthday;
                $r[$t]['imember'] = $y->imember;
                $r[$t]['ishipping_fee'] = $y->ishipping_fee;
                $r[$t]['idiscount_dtl'] = $y->idiscount_dtl;
                $r[$t]['idiscount_percentage_dtl'] = $y->idiscount_percentage_dtl;
                $r[$t]['idiscount_value_dtl'] = $y->idiscount_value_dtl;
                $r[$t]['ibirthday_dtl'] = $y->ibirthday_dtl;
                $r[$t]['imember_dtl'] = $y->imember_dtl;
                $r[$t]['ishipping_dtl'] = $y->ishipping_dtl;
                $r[$t]['isaldo'] = $y->isaldo;
                $r[$t]['itotal_price'] = $y->itotal_price;
                $r[$t]['istock'] = $y->istock;
                $r[$t]['ipoint'] = $y->ipoint;
                $r[$t]['itotal_point'] = $y->itotal_point;
                $r[$t]['iquantity'] = $y->iquantity;
                $r[$t]['iquantity_retur'] = $y->iquantity_retur;
            }
```

Gambar 3. 149 Master Controller Accept return (I)

Ketika approval status bernilai 1 maka proses untuk pengurangan point, penambahan item dan saldo akan berubah. *Developer* membuat variabel dengan nama retur_point yang memanggil dari *table* retur

dengan menjoin beberapa *table* seperti transaction_hdr, item_hdr, users, member, transaction _dtl dengan 2 kondisi where yaitu approval status bernilai 1 dan id_transactiondtl sesuai dengan *reqeust* dari user yang berlaku sebagai admin serta variabel ini menggunakan *method* GET. Setelah itu, *developer* melakukan *foreach* dari *table* retur_point kedalam *array* seperti gambar 3.149 ketika *developer* membutuhkan semua *variable* tersebut untuk melakukan pengurangan point, penambahan saldo, dan penambahan item.

```
$tsetting_point = DB::table('tglobalsetting')->where('vname', 'calc_point')->first();
                $min_point = $tsetting_point->ivalue;

                //$member = DB::table('tmember')->where('id_user', Auth::user()->id_user)->first();

                $update_point = $y->ipoint - ((($y->iprice  / $y->iquantity) / $min_point) * $y->iquantity_retur);
                //dd($update_point);

                $update_price = $y->iprice - (($y->iprice / $y->iquantity) * $y->iquantity_retur);
                //dd($update_price);

                $return_disc = ($y->idiscount_dtl / $y->iquantity) * $y->iquantity_retur;
                //dd($return_disc);
                $return_disc_percentage = ($y->idiscount_percentage_dtl / $y->iquantity) * $y->iquantity_retur;
                //dd($return_disc_percentage);
                $return_disc_value = ($y->idiscount_value_dtl / $y->iquantity) * $y->iquantity_retur;
                //dd($return_disc_value);
                $return_member = ($y->imember_dtl / $y->iquantity) * $y->iquantity_retur;
                //dd($return_member);
                $return_birthday = ($y->ibirthday_dtl / $y->iquantity) * $y->iquantity_retur;
                //dd($return_birthday);
                $total_disc = ($return_disc + $return_disc_percentage + $return_disc_value + $return_member + $return_birthday);
                //dd($total_disc);
```

Gambar 3. 150 Master Controller Accept return (II)

Developer juga membuat beberapa *variable* untuk memudahkan *developer* untuk melakukan penambahan saldo, dan penguranagn point. Variabel tersebut dapat dilihat pada gambar 3.150. *Developer* memanggil value minimal point dari table global setting. *Developer* juga membuat *variabel* update _point untuk menghitung point yang akan dikembalikan dari jumlah item yang diretur, lalu ada variabel update_price untuk menghitung harga item yang dikembalikan. Begitu juga dengan variable return disc, return disc percentage, return disc value, return member, return birthday, dan total disc yang berisikan hasil penjumlahan dari keseluruhan discount yang didapatkan user ketika *return product*.

```
$updateprice_hdr = DB::table('ttransaction_dtl')
                    ->where('iprice', $y->iprice)
                    ->get(['iprice']);

                //dd($updateprice_hdr);
                $updatesaldo = DB::table('ttransaction_dtl')
                    ->where('iprice', $y->iprice)
                    ->get(['iprice']);

                $hdr_price = [];
                $saldo = [];

                $jumlah_saldo = (($y->iprice / $y->iquantity) * $y->iquantity_retur) - $total_disc;
                //dd($jumlah_saldo);

                foreach ($updatesaldo as $o => $p) {
                    $saldo[$o]['iprice'] = $p->iprice;
                }

                foreach ($saldo as $count) {
                    DB::table('tmember')
                        ->where('id_user', $y->id_user)
                        ->where('istatus_member', 1)->update([
                            'isaldo' => $y->isaldo + $jumlah_saldo
                        ]);
                }

                $update_totalprice = $y->itotal_price - $jumlah_saldo;
                //dd($update_totalprice);

                foreach ($updateprice_hdr as $f => $g) {
                    $hdr_price[$f]['iprice'] = $g->iprice;
                }

                foreach ($hdr_price as $count) {
                    DB::table('ttransaction_hdr')
                        ->where('id_transaction', $y->id_transaction)
                        ->update([
                            'itotal_price' => $update_totalprice,
                            'idiscount' => $y->idiscount - $return_disc,
                            'idiscount_percentage' => $y->idiscount_percentage - $return_disc_percentage,
                            'idiscount_value' => $y->idiscount_value - $return_disc_value,
                            'ibirthday' => $y->ibirthday - $return_birthday,
                            'imember' => $y->imember - $return_member,
                        ]);
                }
```

Gambar 3. 151 Code Penambahan Saldo dan Perbarui data pada table transaction_hdr

Setelah membuat *variable* maka *developer* membuat *variable* updateprice_hdr yang mengambil data dari *table* transaction_dtl dengan kondisi price sesuai dengan *array* yang telah dibuat sebelumnya dan menggunakan *method* GET untuk mengambil kolom iprice. Begitu juga dengan updatesaldo. Berikutnya *developer* membuat *variable* jumlah saldo yang berisikan perhitungan dari harga barang dikali dengan quantity yang diretur dan dikurangin *variable* total disc. *Variable* jumlah saldo ini adalah jumlah saldo yang dikembalikan. *Developer* melakukan *foreach* untuk membentuk *array* kolom iprice dari *variable* updatesaldo.

Setelah itu, *developer* juga membuat *foreach* dari *array* saldo untuk mengupdate kolom saldo dari *table* member sesuai dengan id_user yang merequest.

Developer membuat *variable* update_totalprice untuk menghitung dari total price pembelian dikurangi dengan jumlah saldo. *Variable* ini menjadi total price terbaru di table tranasction_hdr. Caranya juga sama dengan melakukan *foreach* pada variable updateprice_hdr untuk mendapatkan *array* dari kolom iprice lalu melakukan *foerach* kembali untuk update pada *table* transaction_hdr. Kolom yang terupdate yaitu total price, discount dari *array* discount di variable retur point dikurangi variable return disc, discount percentage dari array discount percentage di *variable* retur point dikurangi variable return disc percentage, discount value dari *array* discount value di *variable* retur point dikurangi *variable* return disc value, discount birthday dari array discount birthday di *variable* retur point dikurangi *variable* return birthday, dan discount member dari array discount member di *variable* retur point dikurangi *variable* return member dapat dilihat pada gambar 3.151.

```php
if ($r != null) {
                    if ($y->ipoint != 0) {
                        DB::table('ttransaction_dtl')->where('id_transactiondtl', $request->id)
                            ->update([
                                'ipoint' => $update_point,
                                'iquantity' => $y->iquantity - $y->iquantity_retur,
                                'ttransaction_dtl.iprice' => $update_price,
                                'idiscount_dtl' => $y->idiscount_dtl - $return_disc,
                                'idiscount_percentage_dtl' => $y->idiscount_percentage_dtl - $return_disc_percentage,
                                'idiscount_value_dtl' => $y->idiscount_value_dtl - $return_disc_value,
                                'ibirthday_dtl' => $y->ibirthday_dtl - $return_birthday,
                                'imember_dtl' => $y->imember_dtl - $return_member,

                            ]);
                    } else {
                        DB::table('ttransaction_dtl')->where('id_transactiondtl', $request->id)
                            ->update([
                                'ipoint' => $update_point = 0,
                                'iquantity' => $y->iquantity - $y->iquantity_retur,
                                'ttransaction_dtl.iprice' => $update_price,
                                'idiscount_dtl' => $y->idiscount_dtl - $return_disc,
                                'idiscount_percentage_dtl' => $y->idiscount_percentage_dtl - $return_disc_percentage,
                                'idiscount_value_dtl' => $y->idiscount_value_dtl - $return_disc_value,
                                'ibirthday_dtl' => $y->ibirthday_dtl - $return_birthday,
                                'imember_dtl' => $y->imember_dtl - $return_member,
                            ]);
                    }
```

Gambar 3. 152 Code Pengurangan Point dan Perbarui data pada table transaction_dtl

Setelah melakukan *update* pada *table* transaction_hdr maka berikutnya *developer* melakukan *update* point dan data pada *table* transaction _dtl. *Developer* membaut sebuah 2 kondisi IF bahwa *array* tidak boleh *null* dan jika point dari *array* point di *variable* retur point tidak bernilai 0 maka akan mengupdate table transaction_dtl yaitu point dari *variable* point yang telah dibuat sebelumnya lalu quantity akan dikurangi dengan quantity product retur. Untuk data dari kolom discount, discount percentage, discount value, discount member, dan discount birthday melakukan update hampir sama dengan transaction_hdr hanya *variable* yang dikurangi berasal dari data transaction_dtl seperti pada gambar 3.152.

```php
$updatepoint_hdr = DB::table('ttransaction_dtl')
    ->where('id_transaction', $y->id_transaction)
    ->get(['ipoint']);

$hdr_point = [];

$count_point = 0;

foreach ($updatepoint_hdr as $m => $b) {
    $hdr_point[$m]['ipoint'] = $b->ipoint;
}
foreach ($hdr_point as $count) {
    DB::table('ttransaction_hdr')
        ->where('id_transaction', $y->id_transaction)
        ->update([
            'itotal_point' =>  $count_point += $count['ipoint'],
        ]);
}
```

Gambar 3. 153 Update Point di transaction_hdr

Perubahan juga terjadi pada point transaction_hdr maka dari itu *developer* juga perlu mengupdate point dari *table* transaction_hdr. *Developer* membuat sebuah variabel updatepoint hdr dengan mengambil data dari *table* transaction_dtl dengan kondisi id_ transaction sesuai dengan id transaction dengan *method* GET untuk mengambil kolom ipoint. *Developer* juga membuat variabel count point untuk menampung

point terbaru. Setelah itu, *developer* melalukan *foreach* dari variable updatepoint hdr untuk mendapatkan data dari kolom point dan melakukan *foreach* dari *array* hdr_point untuk mengupdate point dari table transaction_hdr dari penjumlahkan point dari *table* transaction_dtl seperti pada gambar 3.153.

```
$updateitem_hdr = DB::table('ttransaction_dtl')
                    ->where('id_item', $y->id_item)
                    ->get(['iquantity']);

                    $hdr_item = [];

                    foreach ($updateitem_hdr as $n => $c) {
                        $hdr_item[$n]['iquantity'] = $c->iquantity;
                    }
                    foreach ($hdr_item as $count) {
                        DB::table('titem_hdr')
                            ->where('id_item', $y->id_item)
                            ->update([
                                'istock' =>  $y->istock + $y->iquantity_retur,
                            ]);
                    }
                }
```

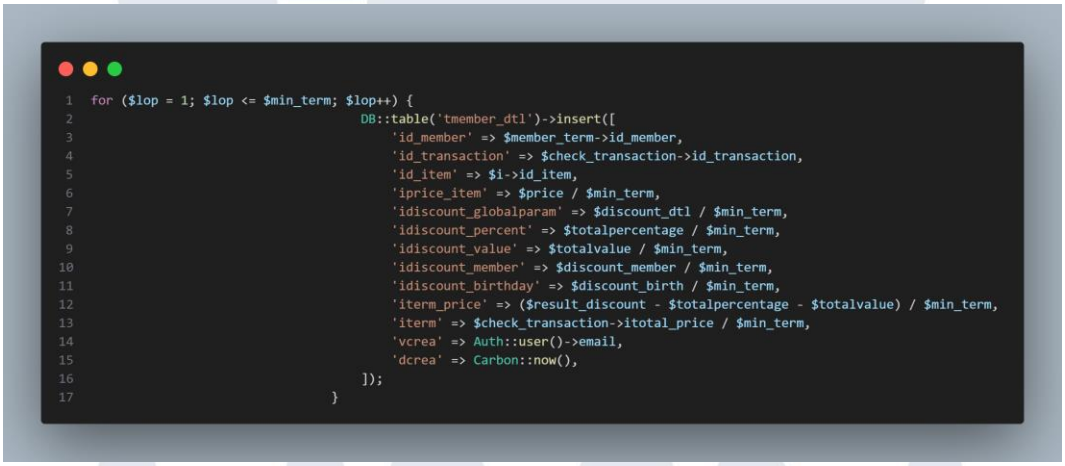
Gambar 3. 154 Penambahan stock pada table item_hdr

Setelah melakukan penambahan saldo dan pengurangan saldo maka berikutnya *developer* melakukan penambahan stock *table* item. *Developer* juga membuat sebuah *variable* dengan nama updateitem_hdr yang mengambil data dari *table* transaction_dtl dengan kondisi where dari id_item dan menggunkan *method* GET untuk mendapatkan kolom quantity. Berikutnya *developer* melakukan *foreach* pada variabel tersebut untuk mendapatkan data dari kolom quantity dan melakukan *foreach* kembali dari variabel *array* hdr_item dengan mengupdate table item_hdr dengan kondisi id_item sama dengan id_item yang diretur. Update terjadi pada kolom stock dengan stock ditambahkan dari quantity product yang diretur oleh user seperti gambar 3.154.

2) Perbaikan Payterm diakibatkan Return Item

a) 19 Oktober – 21 Oktober

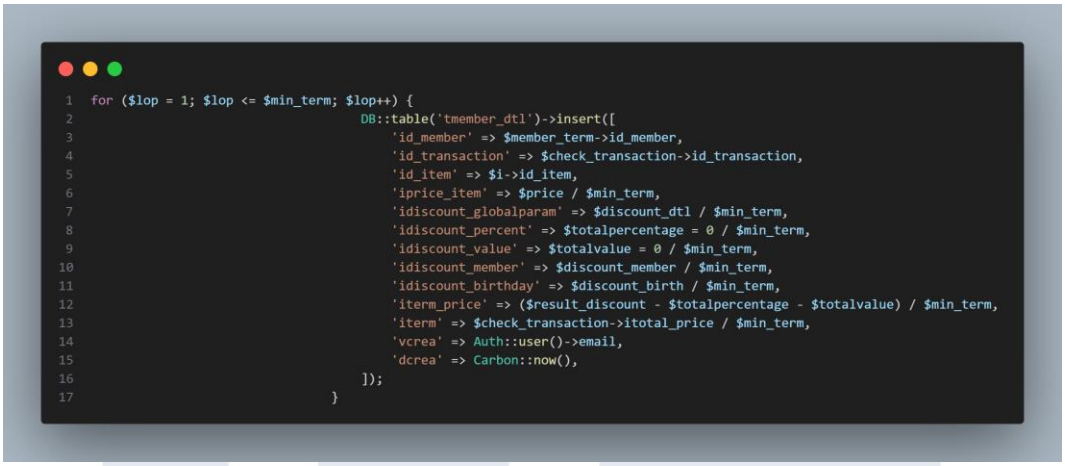
Pada kegiatan ini, *developer* melakukan perbaikan *payterm* dikarenakan terdapat fitur return maka *payterm* ini juga mengalami perubahan yang sebelumnya hanya total price dari transaction_hdr maka sekarang ini perubahan bahwa data *payterm* sesuai dengan item dan pada table member_dtl juga mengalami penambahan beberapa kolom seperti nama item, discount_globalparam, discount percentage, discount value, discount member, discount birthday, harga per item, dan harga term per item.



```
for ($lop = 1; $lop <= $min_term; $lop++) {
                                    DB::table('tmember_dtl')->insert([
                                        'id_member' => $member_term->id_member,
                                        'id_transaction' => $check_transaction->id_transaction,
                                        'id_item' => $i->id_item,
                                        'iprice_item' => $price / $min_term,
                                        'idiscount_globalparam' => $discount_dtl / $min_term,
                                        'idiscount_percent' => $totalpercentage / $min_term,
                                        'idiscount_value' => $totalvalue / $min_term,
                                        'idiscount_member' => $discount_member / $min_term,
                                        'idiscount_birthday' => $discount_birth / $min_term,
                                        'iterm_price' => ($result_discount - $totalpercentage - $totalvalue) / $min_term,
                                        'iterm' => $check_transaction->itotal_price / $min_term,
                                        'vcrea' => Auth::user()->email,
                                        'dcrea' => Carbon::now(),
                                    ]);
                                }
```

Gambar 3. 155 Code Payterm pada item yang mendapatkan discount event

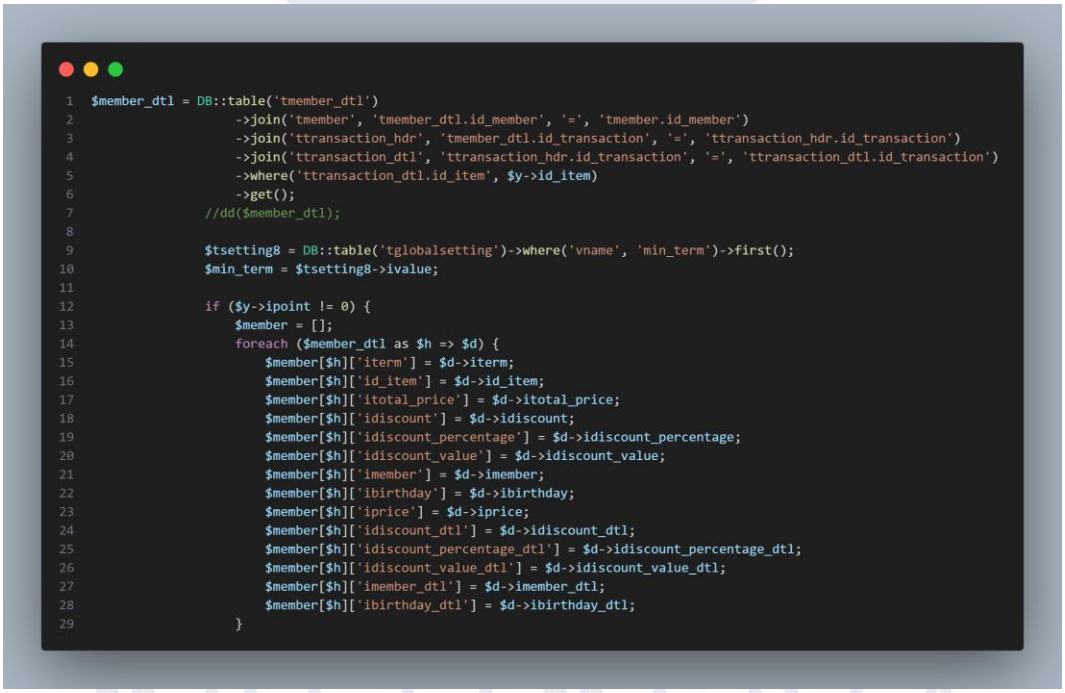
Oleh karena itu, ada perubahan code pada *xendit controller* bagian member yang memiliki status bernilai 1. Gambar 3.155 merupakan *looping* untuk *insert* data ke table member_dtl ketika item mendapatkan discount event.



```
for ($lop = 1; $lop <= $min_term; $lop++) {
                                    DB::table('tmember_dtl')->insert([
                                        'id_member' => $member_term->id_member,
                                        'id_transaction' => $check_transaction->id_transaction,
                                        'id_item' => $i->id_item,
                                        'iprice_item' => $price / $min_term,
                                        'idiscount_globalparam' => $discount_dtl / $min_term,
                                        'idiscount_percent' => $totalpercentage = 0 / $min_term,
                                        'idiscount_value' => $totalvalue = 0 / $min_term,
                                        'idiscount_member' => $discount_member / $min_term,
                                        'idiscount_birthday' => $discount_birth / $min_term,
                                        'iterm_price' => ($result_discount - $totalpercentage - $totalvalue) / $min_term,
                                        'iterm' => $check_transaction->itotal_price / $min_term,
                                        'vcrea' => Auth::user()->email,
                                        'dcrea' => Carbon::now(),
                                    ]);
                                }
```

Gambar 3. 156 Code Payterm pada item yang tidak mendapatkan discount event

Sementara pada gambar 3.156 adalah *looping* insert data ke *table* member_dtl ketika item tidak mendapatkan discount event. Pada saat melakukan *insert* data maka semua value yang berada di *table* member_dtl dibagi dengan variabel min_term yang telah dibuat sebelumnya.



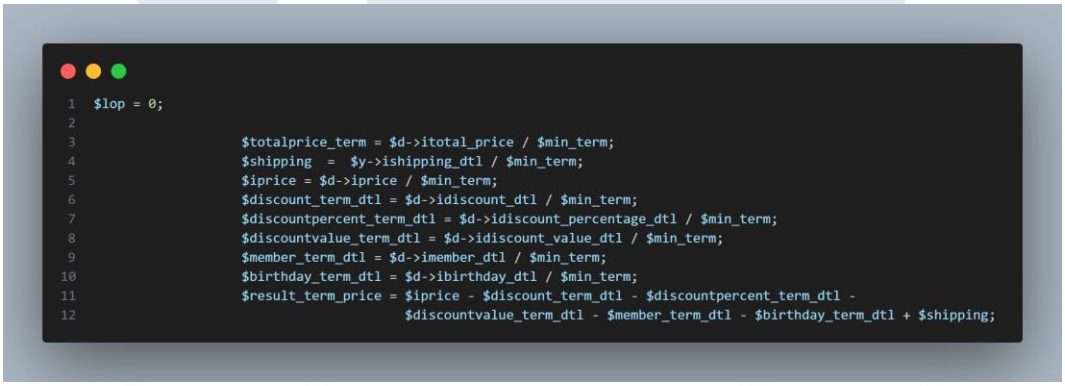
```
$member_dtl = DB::table('tmember_dtl')
                    ->join('tmember', 'tmember_dtl.id_member', '=', 'tmember.id_member')
                    ->join('ttransaction_hdr', 'tmember_dtl.id_transaction', '=', 'ttransaction_hdr.id_transaction')
                    ->join('ttransaction_dtl', 'ttransaction_hdr.id_transaction', '=', 'ttransaction_dtl.id_transaction')
                    ->where('ttransaction_dtl.id_item', $y->id_item)
                    ->get();
                //dd($member_dtl);

                $tsetting8 = DB::table('tglobalsetting')->where('vname', 'min_term')->first();
                $min_term = $tsetting8->ivalue;

                if ($y->ipoint != 0) {
                    $member = [];
                    foreach ($member_dtl as $h => $d) {
                        $member[$h]['iterm'] = $d->iterm;
                        $member[$h]['id_item'] = $d->id_item;
                        $member[$h]['itotal_price'] = $d->itotal_price;
                        $member[$h]['idiscount'] = $d->idiscount;
                        $member[$h]['idiscount_percentage'] = $d->idiscount_percentage;
                        $member[$h]['idiscount_value'] = $d->idiscount_value;
                        $member[$h]['imember'] = $d->imember;
                        $member[$h]['ibirthday'] = $d->ibirthday;
                        $member[$h]['iprice'] = $d->iprice;
                        $member[$h]['idiscount_dtl'] = $d->idiscount_dtl;
                        $member[$h]['idiscount_percentage_dtl'] = $d->idiscount_percentage_dtl;
                        $member[$h]['idiscount_value_dtl'] = $d->idiscount_value_dtl;
                        $member[$h]['imember_dtl'] = $d->imember_dtl;
                        $member[$h]['ibirthday_dtl'] = $d->ibirthday_dtl;
                    }
```

Gambar 3. 157 Payterm pada Master controller

Setelah melakukan perubahan pada *xendit controller* maka ada penambahan code untuk *payterm* di *master controller* ketika user

berlaku sebagai admin *accept* retur product karena ketika user melakukan *return product* maka *payterm* terhadap user tersebut juga mengalami perubahan data. Pertama-tama, *developer* membuat sebuah variabel member_dtl untuk mengambil data dari *table* member_dtl dengan mengabunggkan bebrapa *table* seperti member, transaction_hdr, transaction _dtl dan menggunakan kondisi where dari transaction_dtl berupa id_item yang sesuai dengan id_item yang diretur.

Developer juga mengambil value min_term dari *table* global setting kedalam *variable* min_term. Berikutnya user melakukan *foreach* terhadap variabel tersebut agar *developer* bisa mengambil data dari kolom yang dibutuhkan developer seperti pada gambar 3.157.



```
$lop = 0;

                $totalprice_term = $d->itotal_price / $min_term;
                $shipping  =  $y->ishipping_dtl / $min_term;
                $iprice = $d->iprice / $min_term;
                $discount_term_dtl = $d->idiscount_dtl / $min_term;
                $discountpercent_term_dtl = $d->idiscount_percentage_dtl / $min_term;
                $discountvalue_term_dtl = $d->idiscount_value_dtl / $min_term;
                $member_term_dtl = $d->imember_dtl / $min_term;
                $birthday_term_dtl = $d->ibirthday_dtl / $min_term;
                $result_term_price = $iprice - $discount_term_dtl - $discountpercent_term_dtl -
                                     $discountvalue_term_dtl - $member_term_dtl - $birthday_term_dtl + $shipping;
```

Gambar 3. 158 Variable terbaru payterm

Setelah melakukan *foreach* maka *developer* membuat beberapa variabel yaitu totalprice_term, shipping, iprice, discount term dtl, discount percent term dtl, discount value term dtl, member term dtl, birthday term dtl untuk menampung value yang berada di table transaction_hdr dan transaction_dtl terbaru yang dibagi dengan *variable* min_term. *Developer* juga membuat *variable* result term price untuk menampung value pengurangan dari *variable* yang telah dibuat seperti gambar 3.158.

```
for ($lop = 1; $lop <= $min_term; $lop++) {
                        DB::table('tmember_dtl')
                            ->where('id_transaction', $y->id_transaction)
                            ->where('id_item', $y->id_item)
                            ->update([
                                'iprice_item' => $iprice,
                                'idiscount_globalparam' => $discount_term_dtl,
                                'idiscount_percent' => $discountpercent_term_dtl,
                                'idiscount_value' => $discountvalue_term_dtl,
                                'idiscount_member' => $member_term_dtl,
                                'idiscount_birthday' => $birthday_term_dtl,
                                'iterm_price' => $result_term_price,
                            ]);
                    }
                    for ($lop = 1; $lop <= $min_term; $lop++) {
                        DB::table('tmember_dtl')
                            ->where('id_transaction', $y->id_transaction)
                            ->update([
                                'iterm' => $totalprice_term,

                            ]);
                    }
```

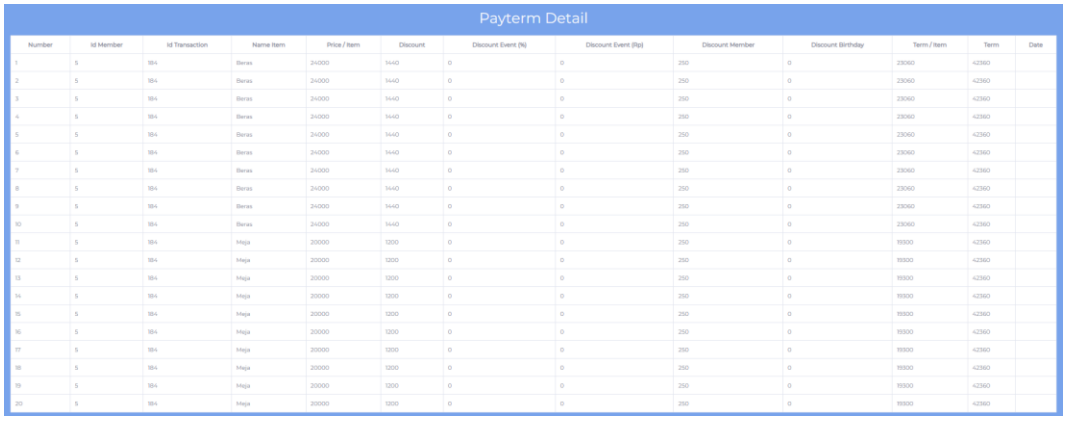
Gambar 3. 159 Looping update table member_dtl

Beriktunya seteleah *developer* membuat variabel maka sekarang ini *developer* melakukan *looping* untuk mengupdate data pada table member_dtl. *Developer* membuat 2 kali *looping* dengan update yang berbeda. Pertama *developer* melakukan *looping* dengan kondisi where id_trransaction dan id_item sesuai dengan id_transaction dan id_item dari product yang diretur dengan mengupdate data pada kolom harga barang, discount, discount event, discount member, discount birthday, dan harga term per item. Sementara *looping* yang kedua ini untuk mengupdate total keseluruhan term baik item yang terkena dampak retur atupun yang tidak dengan menggunakan kondisi where id_transaction sesuai dengan id_transaction ketika return product seperti gambar 3.159.

```
@foreach($master as $i => $u)
        <tr>
          <td>{{++$i}}</td>
          <td>{{$u->id_member}}</td>
          <td>{{$u->id_transaction}}</td>
          <td>{{$u->vname_item}}</td>
          <td>{{$u->iprice_item}}</td>
          <td>{{$u->idiscount_globalparam}}</td>
          <td>{{$u->idiscount_percent}}</td>
          <td>{{$u->idiscount_value}}</td>
          <td>{{$u->idiscount_member}}</td>
          <td>{{$u->idiscount_birthday}}</td>
          <td>{{$u->iterm_price}}</td>
          <td>{{$u->iterm}}</td>
          <td>{{$u->dterm}}</td>
        @endforeach
```

Gambar 3. 160 Perubahan pada HTML member_detail

Perubahan juga terjadi pada tampilan member detail dengan menambahkan beberapa kolom dari variable master yaitu nama item, harga per item, discount, discount event percentage, discount event value, discount member, discount birthday, dan harga term per item seperti pada gambar 3. 160.



Gambar 3. 161 Tampilan Payterm sebelum Return Product

Hasil tampilan payterm sebelum mengalami return product dapat terlihat pada gambar 3.161

Gambar 3. 162 Tampilan Payterm setelah Return Product

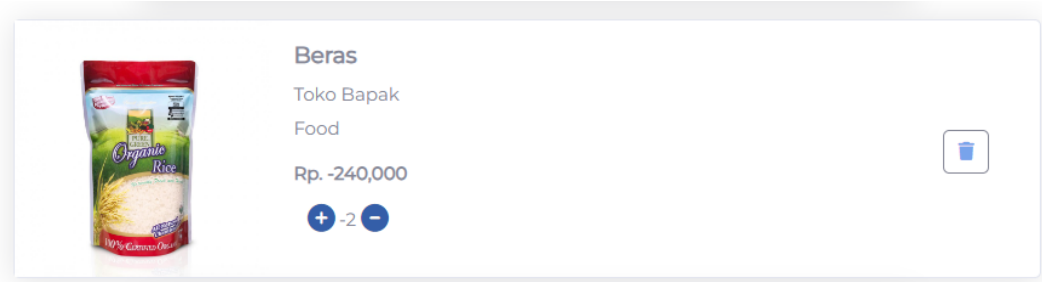
Sementara pada gambar 3.162 adalah hasil tampilan setelah mengalami return product.

3.2.2.4 Bug Fixing

1) Perbaikan Quantity cart, Search Bar, dan Return Transaction

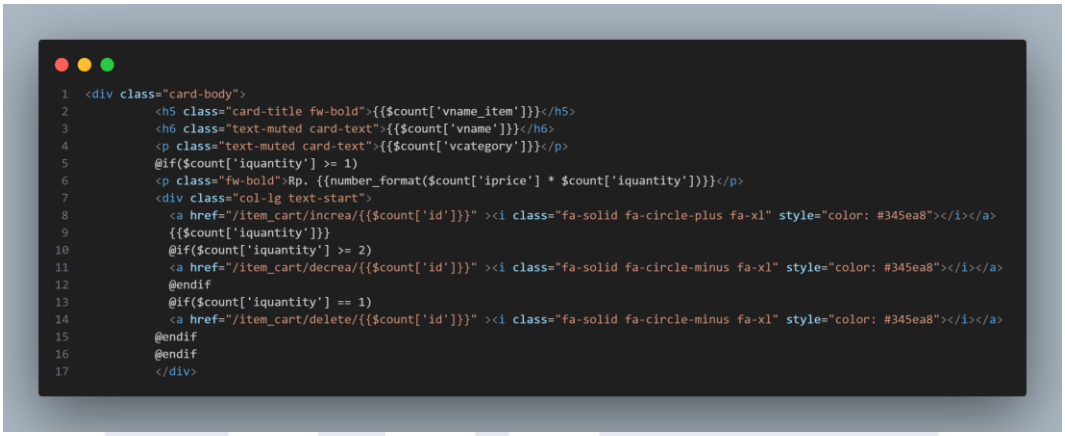
a) 23 Oktober – 27 Oktober

(1) Quantity cart



Gambar 3. 163 Tampilan Bug Quantity Cart

Pada kegiatan ini, *developer* melakukan perbaikan pada quantity cart karena user dapat mengklik icon decrement hingga menjadi quantity di product -1 dan lebih seperti pada gambar 3.163 maka dari itu *developer* memperbaiki bug tersebut agar user hanya dapat mengurangi quantity item sebatas 1 saja dan ketika user mengurangi quantity item menjadi 0 maka item tersebut akan langsung terhapus dari keranjang.



```
<div class="card-body">
          <h5 class="card-title fw-bold">{{$count['vname_item']}}</h5>
          <h6 class="text-muted card-text">{{$count['vname']}}</h6>
          <p class="text-muted card-text">{{$count['vcategory']}}</p>
          @if($count['iquantity'] >= 1)
          <p class="fw-bold">Rp. {{number_format($count['iprice'] * $count['iquantity'])}}</p>
          <div class="col-lg text-start">
            <a href="/item_cart/increa/{{$count['id']}}" ><i class="fa-solid fa-circle-plus fa-xl" style="color: #345ea8"></i></a>
            {{$count['iquantity']}}
            @if($count['iquantity'] >= 2)
            <a href="/item_cart/decrea/{{$count['id']}}" ><i class="fa-solid fa-circle-minus fa-xl" style="color: #345ea8"></i></a>
            @endif
            @if($count['iquantity'] == 1)
            <a href="/item_cart/delete/{{$count['id']}}" ><i class="fa-solid fa-circle-minus fa-xl" style="color: #345ea8"></i></a>
          @endif
          @endif
          </div>
```

Gambar 3. 164 HTML Cart Quantity

Oleh karena itu, *developer* menggunakan kondisi IF untuk bisa menvalidasi kondisi tersebut. Pertama-tama *developer* membuat sebuah kondisi jika quantity bernilai lebih dari sama dengan 1 maka akan memunculkan 2 icon tambah dan icon kurang. Dalam kondisi tersebut terdapat 2 kondisi IF jika quantity lebih dari 2 maka tombol icon kurang akan mengurangi item tapi ketika quantity bernilai 1 maka icon kurang tersebut akan berubah fungsi untuk menghapus item dari tampilan cart dapat terlihat pada gambar 3.164.

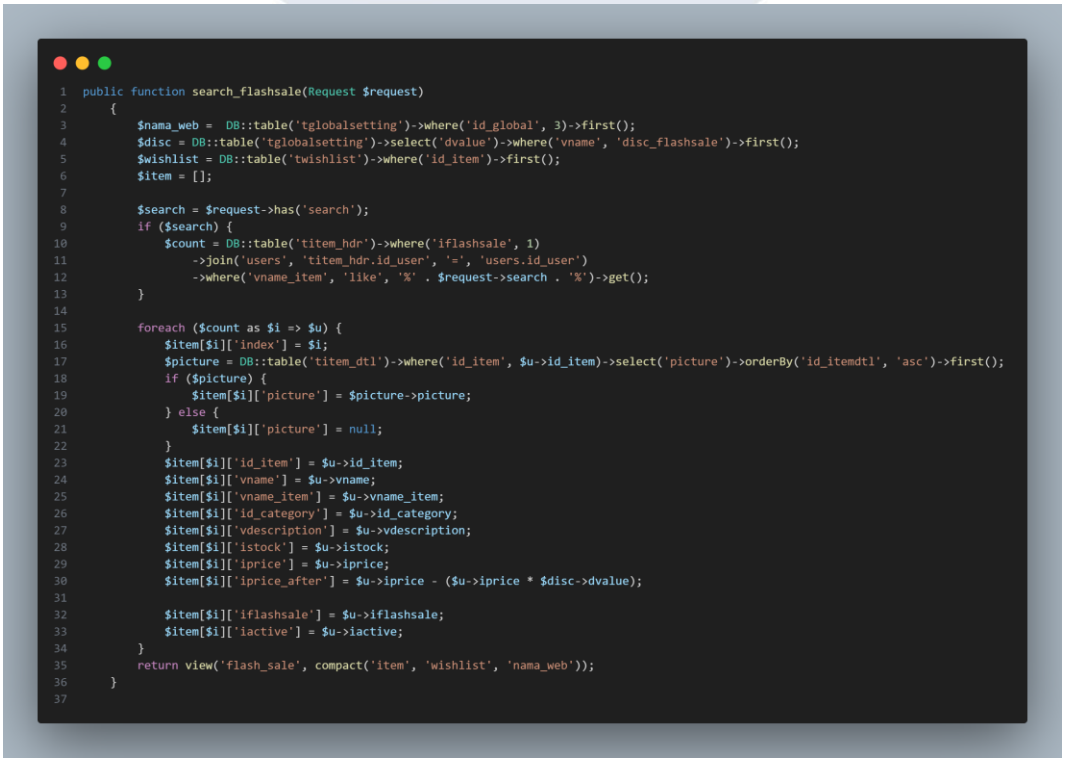
(2) Search Bar

Pada kegiatan ini *developer* melakukan perbaikan dan penambahan pada fitur search di beberapa tampilan yaitu tampilan flash sale, product more dan return trasnaction. Dengan adanya Search abr ini bisa membantu users untuk mencari nama item atau nomor transaction pada website e-commerce.

```
<!-- Search Bar -->
    <div class="row justify-content-center mb-3">
      <div class="col-md-5">
        <form action="/flash_sale/search" method="get">
          <div class="input-group rounded">
            <input type="search" class="form-control rounded" placeholder="Search here..." name="search" value="{{request('search')}}" />
            <button class="input-group-text border-0" id="search-addon" type="submit">
              <i class="fa fa-search"></i>
            </button>
          </div>
        </form>
      </div>
    </div>
```

Gambar 3. 165 Form Search Bar Flash Sale

Pertama-tama *developer* melakukan penambahan form pada HTML tampilan flashsale agar bisa menampilkan search bar. Lalu pada form menambahkan action yang berisikan URL dari tampilan search dengan *method* GET. *Developer* juga membuat *name* dari input tersebut dengan search dan value dari *request* search. Value ini berguna untuk ketika user mencari nama item maka nama item tersebut masih ada di search bar seperti pada gambar 3.165.



```
public function search_flashsale(Request $request)
    {
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        $disc = DB::table('tglobalsetting')->select('dvalue')->where('vname', 'disc_flashsale')->first();
        $wishlist = DB::table('twishlist')->where('id_item')->first();
        $item = [];

        $search = $request->has('search');
        if ($search) {
            $count = DB::table('titem_hdr')->where('iflashsale', 1)
                ->join('users', 'titem_hdr.id_user', '=', 'users.id_user')
                ->where('vname_item', 'like', '%' . $request->search . '%')->get();
        }

        foreach ($count as $i => $u) {
            $item[$i]['index'] = $i;
            $picture = DB::table('titem_dtl')->where('id_item', $u->id_item)->select('picture')->orderBy('id_itemdtl', 'asc')->first();
            if ($picture) {
                $item[$i]['picture'] = $picture->picture;
            } else {
                $item[$i]['picture'] = null;
            }
            $item[$i]['id_item'] = $u->id_item;
            $item[$i]['vname'] = $u->vname;
            $item[$i]['vname_item'] = $u->vname_item;
            $item[$i]['id_category'] = $u->id_category;
            $item[$i]['vdescription'] = $u->vdescription;
            $item[$i]['istock'] = $u->istock;
            $item[$i]['iprice'] = $u->iprice;
            $item[$i]['iprice_after'] = $u->iprice - ($u->iprice * $disc->dvalue);

            $item[$i]['iflashsale'] = $u->iflashsale;
            $item[$i]['iactive'] = $u->iactive;
        }
        return view('flash_sale', compact('item', 'wishlist', 'nama_web'));
    }
```

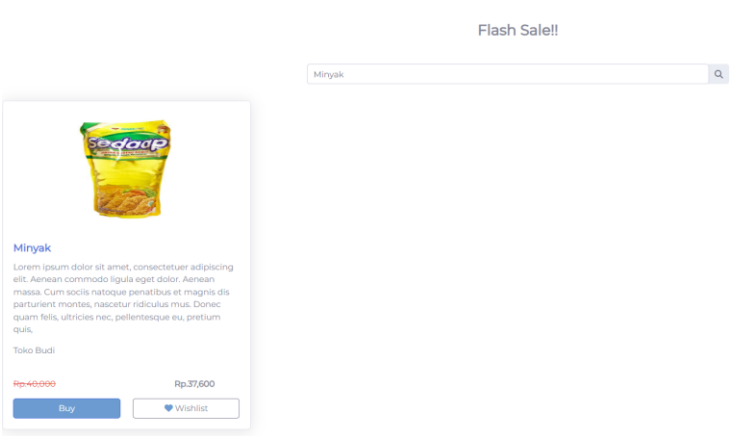
Gambar 3. 166 Function Search Flash Sale

Lalu *developer* juga menambahkan *function* baru dengan nama search_flashsale isi code nya hampir sama dengan *function* flashsale hanya yang membedakan yaitu adanya penambahan code untuk bisa mencari nama item. *Developer* membuat sebuah variabel search yang berisikan *request* dari nama input form yaitu search. Jika terdapat nilai dari search maka terdapat *variable* count dari *function* flashsale yang membedakan yaitu terdapat kondisi where yang berguna untuk mencari nama item yang sesuai dengan request search yang diinput oleh users ketika mencari nama item lalu menggunakan *method* GET seperti gambar 3.166.

```
1  Route::get('flash_sale/search', 'App\Http\Controllers\UserPageController@search_flashsale');
```

Gambar 3. 167 Route Search Flash Sale

Lalu *developer* membuat sebuah *route* dengan *method* GET dan menggunakan URL yang ada didalam form search di HTML flash_sale dengan memanggil function search_flashsale seperti pada gambar 3.167.



Gambar 3. 168 Tampilan Search bar di Flash Sale

Hasil tampilan search bar pada tampilan flash_sale dapat terlihat seperti pada gambar 3.168. Ketika mengketikan product minyak maka akan muncul pencarian mengenai minyak.

```blade
<!-- Search Bar -->
    <div class="row justify-content-center mb-3">
      <div class="col-md-5">
        <form action="/product/more/search" method="get">
          <div class="input-group rounded">
            <input type="search" class="form-control rounded" placeholder="Search here..." name="search" value="{{request('search')}}" />
            <button class="input-group-text border-0" id="search-addon" type="submit">
              <i class="fa fa-search"></i>
            </button>
          </div>
        </form>
```

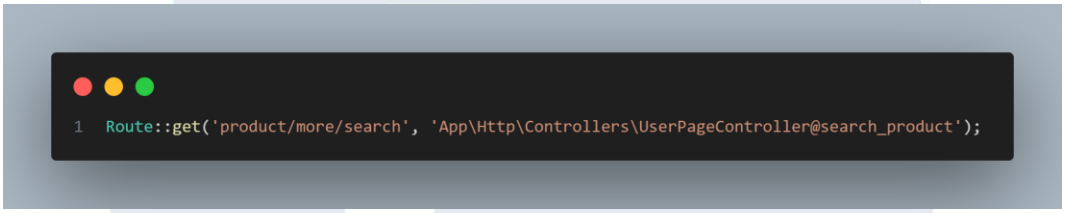
Gambar 3. 169 Form Search Bar Product More

Berikutnya pada product more *developer* juga melakukan penambahan form pada HTML seperti tampilan flashsale hanya dibedakan dengan alamat URL yang digunakan karena ini product more maka URL nya perlu mengembalikan tampilan product more search seperti pada gambar 3.169. *Developer* juga menambahkan *function* baru dengan nama search_product isi code nya hampir sama dengan *function* product_more hanya yang membedakan yaitu adanya penambahan code untuk bisa mencari nama item.

```blade
public function search_product(Request $request)
    {
        $wishlist = DB::table('twishlist')->where('id_item')->first();
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        $search = $request->has('search');
        if ($search) {
            $count = DB::table('titem_hdr')->where('iflashsale', 0)
                ->join('users', 'titem_hdr.id_user', '=', 'users.id_user')
                ->where('vname_item', 'like', '%' . $request->search . '%')->get();
        }

        $item = [];
        foreach ($count as $i => $u) {

            $item[$i]['index'] = $i;
            $picture = DB::table('titem_dtl')->where('id_item', $u->id_item)->select('picture')->orderBy('id_itemdtl', 'asc')->first();
            if ($picture) {
                $item[$i]['picture'] = $picture->picture;
            } else {
                $item[$i]['picture'] = null;
            }
            $item[$i]['id_item'] = $u->id_item;
            $item[$i]['vname'] = $u->vname;
            $item[$i]['vname_item'] = $u->vname_item;
            $item[$i]['id_category'] = $u->id_category;
            $item[$i]['vdescription'] = $u->vdescription;
            $item[$i]['istock'] = $u->istock;
            $item[$i]['iprice'] = $u->iprice;
            // $item[$i]['iprice_after'] = $u->iprice - ($u->iprice * $u->idisc);

            $item[$i]['iflashsale'] = $u->iflashsale;
            $item[$i]['iactive'] = $u->iactive;
        }
        return view('product_more', compact('item', 'wishlist', 'count', 'nama_web'));
    }
```

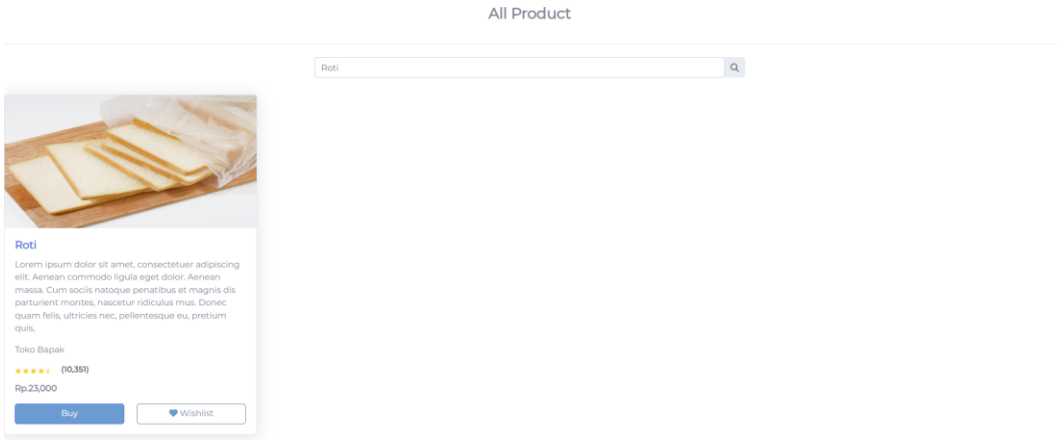
Gambar 3. 170 Function Search Product More

Developer juga membuat sebuah variabel search seperti *function* search_flashsale. Jika terdapat nilai dari search maka terdapat *variable* count dari *function* product_more dan terdapat kondisi where yang berguna untuk mencari nama item yang sesuai dengan *request* search yang diinput oleh user ketika mencari nama item lalu menggunakan *method* GET seperti pada gambar 3.170.



Gambar 3. 171 Route Search Product More

Berikutnya *developer* membuat sebuah *route* dengan *method* GET dan menggunakan URL yang ada didalam form search product_more dengan memanggil *function* search_product seperti pada gambar 3.171.



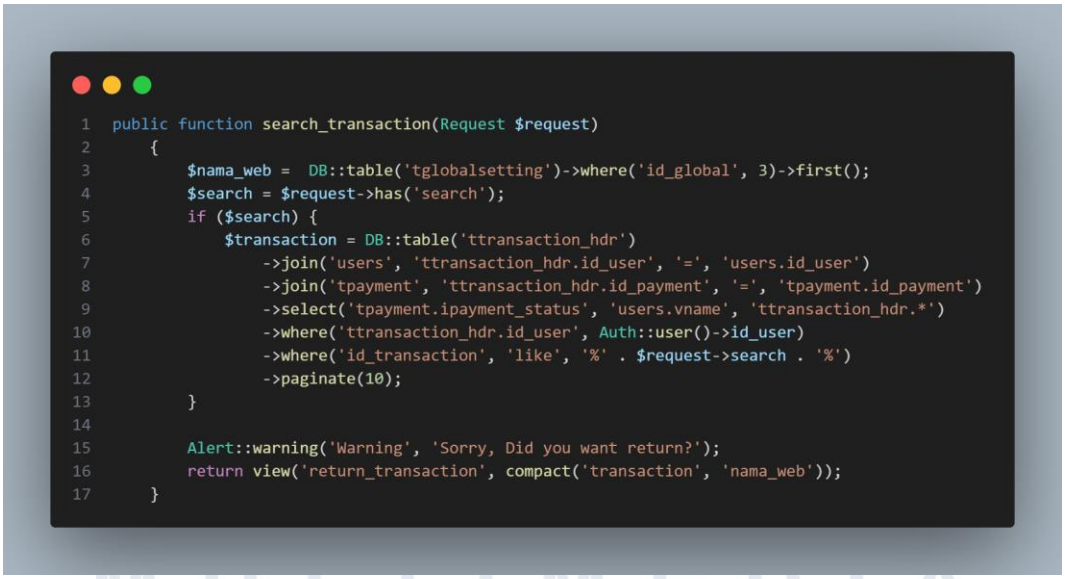
Gambar 3. 172 Tampilan Search bar di Product More

Hasil tampilan search bar pada tampilan product_more akan seperti pada gambar 3.172. Ketika mengketikan product roti maka akan muncul pencarian mengenai roti.

```
<!-- Search Bar -->
<div class="row justify-content-center mb-3">
  <div class="col-md-5">
    <form action="/return_transaction/search" method="get">
      <div class="input-group rounded">
        <input type="search" class="form-control rounded" placeholder="Search here..." name="search" value="{{request('search')}}" />
        <button class="input-group-text border-0" id="search-addon" type="submit">
          <i class="fa fa-search"></i>
        </button>
      </div>
    </form>
  </div>
</div>
```

Gambar 3. 173 Form Search Bar Return Product

Terakhir pada *return product*, *developer* juga melakukan penambahan form pada HTML return transaction seperti tampilan flash_sale dan product_more hanya dibedakan dengan alamat URL yang digunakan karena ini return_transaction maka URL nya perlu mengembalikan tampilan return transaction search seperti pada gambar 3.173. Berikutnya *developer* juga menambahkan *function* baru dengan nama search_ transaction isi dari *function* ini hampir sama dengan *function* index_transaction_return hanya yang membedakan yaitu adanya perubahan code.



```
public function search_transaction(Request $request)
    {
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        $search = $request->has('search');
        if ($search) {
            $transaction = DB::table('ttransaction_hdr')
                ->join('users', 'ttransaction_hdr.id_user', '=', 'users.id_user')
                ->join('tpayment', 'ttransaction_hdr.id_payment', '=', 'tpayment.id_payment')
                ->select('tpayment.ipayment_status', 'users.vname', 'ttransaction_hdr.*')
                ->where('ttransaction_hdr.id_user', Auth::user()->id_user)
                ->where('id_transaction', 'like', '%' . $request->search . '%')
                ->paginate(10);
        }

        Alert::warning('Warning', 'Sorry, Did you want return?');
        return view('return_transaction', compact('transaction', 'nama_web'));
    }
```

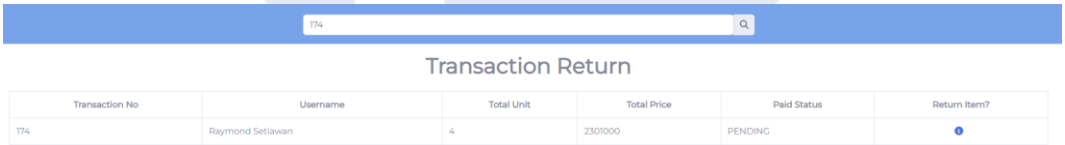
Gambar 3. 174 Function Search Return Product

Developer membuat sebuah variabel search untuk bisa mendapatkan *request* user ketika mencari id transaction. Jika terdapat nilai dari search

maka terdapat *variable* transaction dari *function* index_transaction_return dan terdapat kondisi where yang berguna untuk mencari id transaction yang sesuai dengan *request* search yang diinput oleh user ketika mencari nama item lalu menggunakan *method* GET seperti pada gambar 3.174.

Gambar 3. 175 Route Search Return Product

Berikutnya *developer* membuat sebuah *route* dengan *method* GET dan menggunakan URL yang ada didalam form search return_transaction dengan memanggil *function* search_ transaction seperti pada gambar 3.175.



Gambar 3. 176 Tampilan Search bar di Return Product

Hasil tampilan search bar pada tampilan return_transaction akan seperti pada gambar 3.176. Ketika mengketikan no transaction maka akan muncul pencarian mengenai no transaction tersebut.

(3) Tampilan Return Transaction

```php
<?php

namespace App\Providers;

use Illuminate\Support\ServiceProvider;
use Illuminate\Pagination\Paginator;

class AppServiceProvider extends ServiceProvider
{
    /**
     * Register any application services.
     *
     * @return void
     */
    public function register()
    {
        //
    }

    /**
     * Bootstrap any application services.
     *
     * @return void
     */
    public function boot()
    {
        Paginator::useBootstrap();
    }
}

```

Gambar 3. 177 Penambahan Method Pagination

Setelah *developer* melakukan perbaikan search bar maka berikutnya *developer* melakukan perbaikan pada tampilan return transaction yang belum menggunakan *pagination*. Oleh karena itu, *developer* melakukan perubahan agar tampilan dari return transaction bisa berhalaman. Pertama-tama, *developer* melakukan pemanggilan *method pagination* di AppServiceProvider dengan use Illuminate\Pagination\Paginator lalu menambahkan Paginator::useBootstrap() pada function boot seperti pada gambar 3.177

```php
public function index_transaction_return()
    {
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        $transaction = DB::table('ttransaction_hdr')
            ->join('users', 'ttransaction_hdr.id_user', '=', 'users.id_user')
            ->join('tpayment', 'ttransaction_hdr.id_payment', '=', 'tpayment.id_payment')
            ->select('tpayment.ipayment_status', 'users.vname', 'ttransaction_hdr.*')
            ->where('ttransaction_hdr.id_user', Auth::user()->id_user)
            ->paginate(10);


        Alert::warning('Warning', 'Sorry, Did you want return?');
        return view('return_transaction', compact('transaction', 'nama_web'));
    }
```

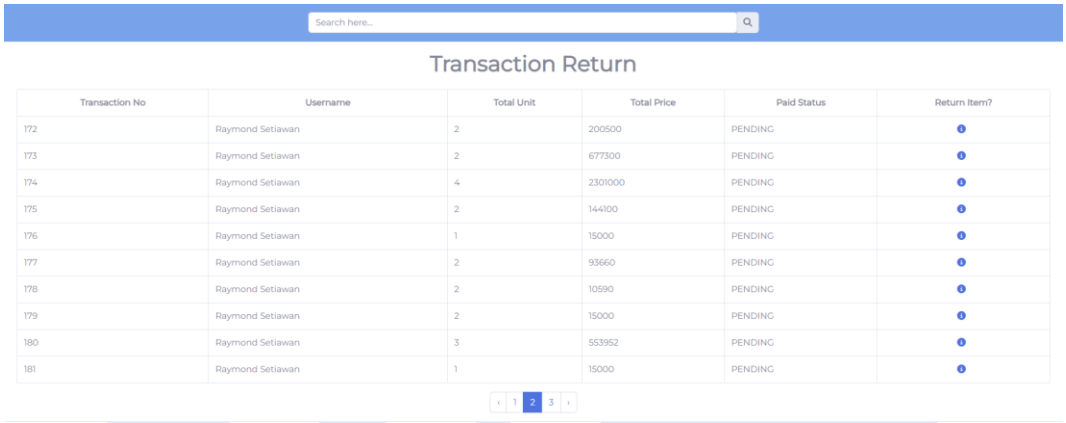
Gambar 3. 178 Perubahan function index_transaction_return

Berikutnya *developer* mengubah *method* dalam mengambil data di *function* index_transaction_return yang sebelumnya menggunakan *method* GET sekarang ini menggunakan *method Paginate* dengan membatasi 1 halam hanya 10 data yang tampil jika lebih akan berpindah ke halaman berikutnya seperti pada gambar 3.178.

```php
<div class="d-flex justify-content-center">
        {{$transaction->onEachSide(1)->links()}}
    </div>

```

Gambar 3. 179 HTML Return Transaction

Setelah itu pada tampilan HTML return trasnaction juga ada penambahan yaitu variabel transaction->onEachSide(1)->links() seperti pada gambar 3.179. *Method* onEachSide(1)->links() untuk mengontrol berapa banyak link tambahan yang akan ditampikan di setiap sisi halaman tampilan.



Gambar 3. 180 Hasil Tampilan Menggunakan Pagination

Hasil tampilan yang menggunakan *pagination* seperti pada gambar 3.180 sehingga data transaction hanya maksimal 10 per halaman. Jika lebih 10 maka akan berpindah pada halaman berikutnya

2) Penambahan Max Return hanya 3 hari

a) 27 Oktober – 27 Oktober

Pada kegiatan ini, *developer* diberikan tugas untuk melakukan penambahan pada *return product* bahwa *return product* hanya bisa dilakukan selama 3 hari setelah *product* tersebut diterima oleh user. Jika lebih dari 3 hari maka user tidak bisa mengembalikan produk. *Developer* melakukan penambahan pada *function* index_retur di *usepagecontroller*

```
public function Index_retur($id)
    {
        //dd($id);
        $nama_web =  DB::table('tglobalsetting')->where('id_global', 3)->first();
        $retur = DB::table('ttransaction_dtl')
            ->join('titem_hdr', 'ttransaction_dtl.id_item', '=', 'titem_hdr.id_item')
            ->join('ttransaction_hdr', 'ttransaction_dtl.id_transaction', '=', 'ttransaction_hdr.id_transaction')
            ->join('users', 'ttransaction_hdr.id_user', '=', 'users.id_user')
            ->where('ttransaction_dtl.id_transaction', $id)
            ->where('ttransaction_hdr.id_user', Auth::user()->id_user)
            ->select('ttransaction_hdr.*', 'titem_hdr.*', 'ttransaction_dtl.*', 'ttransaction_hdr.id_user')->get();

            $tsetting = DB::table('tglobalsetting')->where('vname', 'max_retur')->first();
            $max_retur = $tsetting->ivalue;
            //dd($max_retur);
            $day = [];

            foreach ($retur as $i => $u) {
                $day[$i]['dsarriv'] = $u->dsarriv;
            }
            //dd($y->dsarriv);

            $date_arrive = Carbon::parse($u->dsarriv);
            //dd($date_arrive);
            $date_max_return = Carbon::parse($u->dsarriv)->addDays($max_retur);
            //dd($date_max_return);
            $today = now();
            //dd($today);

           $condition_max= $date_max_return >= $today;
           $condition_date = $date_arrive <= $today;
           $condition_date1 = $date_arrive >= $today;

           $num = 0;
            if (($condition_date|| $condition_date1) && $condition_max) {
                $num= 1;
             }else{
                $num = 2;
             }
             //dd($e);


        Alert::Warning('Choose', 'The item to return!');

        return view('return_item', compact('retur', 'num', 'nama_web'));
    }
```

Gambar 3. 181 Penambahan code untuk max_retur

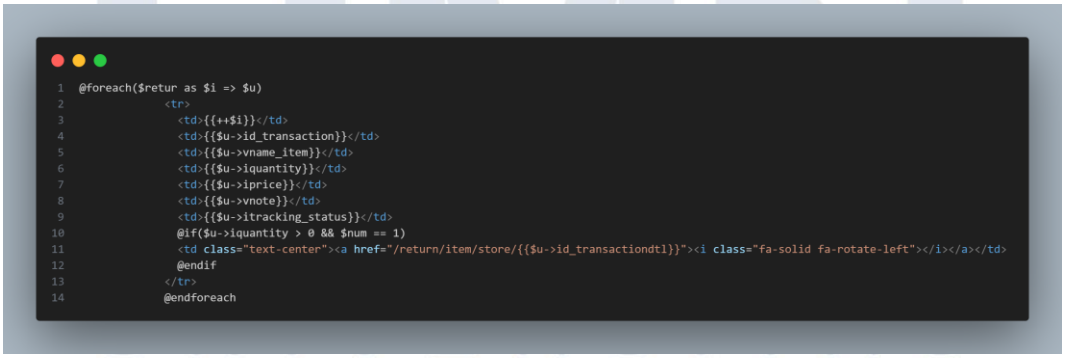
Pertama-tama, *developer* membuat variabel tsetting untuk memanggil data dari global setting yang berisikan max return (jumlah hari yang ditetapkan oleh admin untuk user dalam mengembalikan barang). Data tersebut *developer* memasukan kedalam *variable* max_retur. Berikutnya *developer* melakukan *foreach* dari variabel retur untuk bisa mendapatkan tanggal diterima. Setelah *developer* mendapatkan tanggal diterima maka *developer* membuat sebuah variabel dengan nama date_arrive untuk mengubah format string kedalam bentuk tanggal dengan *method* Carbon ::parse().

Developer juga membuat sebuah variabel date_max_return yang bernilai dari tanggal diterima yang dikonversikan menjadi format

tanggal dan menggunakan *method* addDays sesuai variabel max_return untuk mendapatkan tanggal maksimal user mengembalikan *product*. Berikutnya terdapat variabel today yang bernilai tanggal yang ada di local. Lalu *developer* juga membuat sebuah beberapa *variable* kondisi yaitu:

- $condition_max= $date_max_return >= $today yang berarti variable date_max_return lebih besar sama dengan dari tanggal hari ini akan bernilai true.
- $condition_date = $date_arrive <= $today yang berarti variable date_arrive kurang sama dengan dari tanggal hari ini akan bernilai true.
- $condition_date1 = $date_arrive >= $today yang berarti variable date_arrive lebih besar sama dengan dari tanggal hari ini akan bernilai true.

Dari beberapa kondisi ini maka *developer* membuat 1 kondisi IF jika condition _date atau condition _date_1 benar dan condition_max bernilai benar maka akan mengembalikan nilai 1 pada *variable* num. Jika tidak maka akan mengembalikan nilai num = 2. Nilai tersebut akan dipassing ke tampilan return_item seperti pada gambar 3.181.



```
@foreach($retur as $i => $u)
            <tr>
              <td>{{++$i}}</td>
              <td>{{$u->id_transaction}}</td>
              <td>{{$u->vname_item}}</td>
              <td>{{$u->iquantity}}</td>
              <td>{{$u->iprice}}</td>
              <td>{{$u->vnote}}</td>
              <td>{{$u->itracking_status}}</td>
              @if($u->iquantity > 0 && $num == 1)
              <td class="text-center"><a href="/return/item/store/{{$u->id_transactiondtl}}"><i class="fa-solid fa-rotate-left"></i></a></td>
              @endif
            </tr>
            @endforeach
```

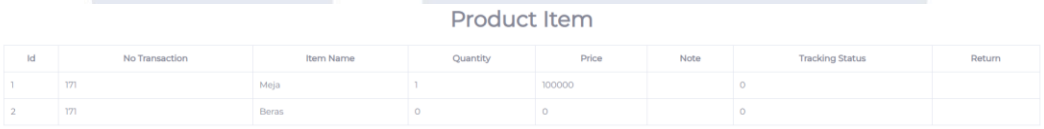
Gambar 3. 182 Perubahan pada HTML Return Item

Setelah berhasil mendapatkan nilai num maka pada tampilan HTML return_item terdapat sebuah kondisi IF seperti pada gambar 3.182 jika

nilai dari variabel num bernilai 1 maka ikon untuk retur masih bisa diklik Namun jika nilai num bernilai 2 maka ikon return tidak akan muncul

Gambar 3. 183 Tampilan Return product ketika kurang dari Max_Return

Pada gambar 3.183 merupakan tampilan return product ketika user masih bisa mengembalikan product yang cacat atau rusak karena belum lebih dari 3 hari dari penerimaan barang tersebut.



Gambar 3. 184 Tampilan Return product ketika lebih dari Max_Return

Sementara, pada gambar 3.184 merupakan tampilan return product ketika user suadh tidak bisa mengembalikan product yang cacat atau rusak karena sudah lebih dari 3 hari dari penerimaan barang tersebut.

3.2.2.5 User Accept Test (UAT)

1) Phase Feedback UAT Result

a) 30 Oktober 2023 - 25 November 2023

Pada kegiatan ini berlangsung, *developer* menunggu hasil dari inputan test yang dilakukan oleh *supervisor* agar memastikan bahwa project yang selama ini dikerjakan telah berfungsi dengan baik dan tidak mengalami *bug* atau *error* jika mengalami *error* maka diperlukan perbaikan hingga tidak terdapat *bug*. Selama fase ini *developer* tidak mendapatkan *feedback* dari *supervisor* untuk memperbaiki dari tugas yang telah dikerjakan oleh *developer* selama kegiatan magang berlangsung dan terdapat satu rekan kerja kami yang masih terdapat isu. Pada tanggal 25 November 2023, *supervisor* telah melakukan evaluasi

grade kedua pada rekan magang dan kegiatan magang MBKM Track 2 telah berakhir.

3.3 Kendala yang Ditemukan

Selama penulis selaku peserta magang melakukan kegiatan magang MBKM Track 2 mengalami beberapa kendala yang dihadapi oleh penulis ketika melakukan kegiatan magang di PT. Ritzproject Sinergi Visitama. Kendala yang sering dihadapi penulis ketika awal-awal melakukan magang yaitu mengerjakan tugas yang diberikan oleh *supervisor* atau PMO kepada penulis karena penulis masih belum memahami konsep *framework* Laravel yang digunakan dalam melakukan pengembangan website *e-commerce* serta kendala ini juga didukung adanya keterbatasan mengenai informasi penyelesaian yang berada di internet dan terkadang konsep penyelesaian yang berada di internet tidak terlalu sama dengan keinginan penulis. Terdapat kendala juga yaitu ketika mengerjakan *task* secara bersama terdapat beberapa rekan magang yang masih pasif sehingga hal ini membuat pekerjaan task tersebut tidak dihandle secara merata.

Dari kendala yang diatas maka penulis dapat menyimpulkan kendala tersebut kedalam beberapa point sebagai berikut:

1. Penulis mengalami kendala dalam konsep Laravel dikarenakan penulis belum pernah mempelajari Laravel selama perkuliahan sehingga penulis perlu mempelajari konsep Laravel terlebih dahulu. Selain itu konsep Laravel ini juga termasuk baru digunakan oleh perusahan sehingga belum ada orang yang telah expert dalam menggunakan *framework* ini untuk bisa menyelesaikan kendala yang dihadapi oleh penulis.
2. Penulis mengalami kesulitan dalam mendapatkan informasi penyelesaian dari kendala tersebut dikarenakan konsep penyelesaian yang berada di internet sulit dimengerti oleh penulis dan terkadang konsep penyelesaian yang digunakan dari internet juga berbeda dengan konsep yang digunakan

oleh penulis dalam mnegerjakan tugas sehingga beberapa penyelesaian menjadi tidak relate.

3. Pada saat melakukan pekerjaan task secara bersama-sama dengan tim *back-end developer* ketika awal penerimaaan magang terdapat beberapa rekan kerja yang masih pasif sehingga membuat pekerjaan dari task tersebut tidak memiliki kejelasan untuk menghandle task dan membuat proses penyelesaian yang memakan waktu yang lama

3.4 Solusi atas Kendala yang Ditemukan

Dibalik penulis mendapatkan beberapa kendala ketika melakukan kegiatan magang MBKM Track 2 di PT Ritzproject Sinergi Visitama. Penulis memperoleh beberapa solusi dari kendala yang dihadapi untuk mengatasi kendala sebagai berikut:

1. *Supervisor* atau PMO dari PT Ritzproject Sinergi Visitama memberikan sebuah keringanan waktu dalam mengerjakan tugas yang diberikan sehingga penulis bisa mendapatkan waktu untuk mempelajari konsep Laravel dan konsep penyelesaian dari tugas yang diberikan agar penulis dapat memahami konsep penyelesaiannya sehingga penulis dapat secara langsung mengerjakan project *e-commerce* secara maksimal.
2. Penulis juga menyelesaikan kendala dari tugas yang diberikan dengan sering melakukan improvisasi dari beberapa *method* yang telah ada sebelumnya dan menemukan cara penyelesaikan dengan *method* sendiri yang direferensikan dari beberapa forum seperti stackoverflow, laracasts dan menonton beberapa video di youtube.
3. Dari pengalaman task yang dikerjakan secara bersama-sama dengan tim *back-end developer* membuat keputusan bahwa adanya pembagian task untuk setiap anggota *back-end developer* sehingga memiliki kejelasan dan tanggung jawab masing-masing mengenai task yang diberikan oleh PMO atau *supervisor*.