BAB IV
IMPLEMENTASI DAN UJI COBA

4.1 Implementasi

Bagian ini menguraikan pelaksanaan dari rancangan yang sudah dijabarkan sebelumnya. mencakup detail tentang spesifikasi produk, penerapan database, dan pelaksanaan program.

4.1.1 Spesifikasi Produk

Agar program dapat berjalan lancar, aplikasi dari hasil penelitian ini memerlukan spesifikasi sebagai berikut :

1. *Hardware*

Berikut adalah *hardware* yang digunakan untuk menunjang penelitian ini :

a. Intel(R) Core(TM) i5-1035G1 CPU @ 1.00 GHz 1.19 GHz

b. 8 GB

c. SSD 512 GB

e. *Mouse dan Keyboard*

f. *Printer*

2. *Software*

Berikut adalah *software* yang digunakan untuk menunjang penelitian ini :

a. Windows 10 Home Single Language

b. Web browser Chrome & Microsoft edge untuk

c. Web server Xampp

d. MySQL untuk keperluan database

e. Bahasa pemrograman PHP dan Javascript

f. Laravel sebagai Framework

g. Microsoft Visual Studio Code sebagai text editor

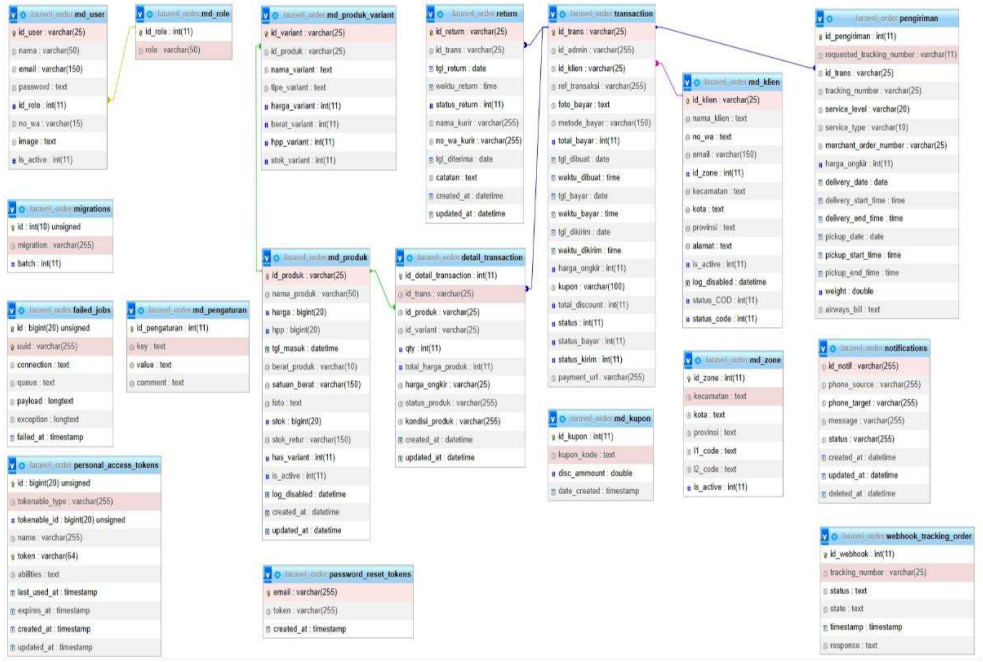
h. API Whatsapp untuk mengirimkan pesan / notifikasi otomatis.

i. Ngrok sebagai webhook

j. Midtrans untuk penunjang metode pembayaran transfer

4.1.2 Implementasi Database

Berikut adalah penerapan rancangan database yang sudah di gambarkan pada bab sebelumnya ke dalam mySQL.



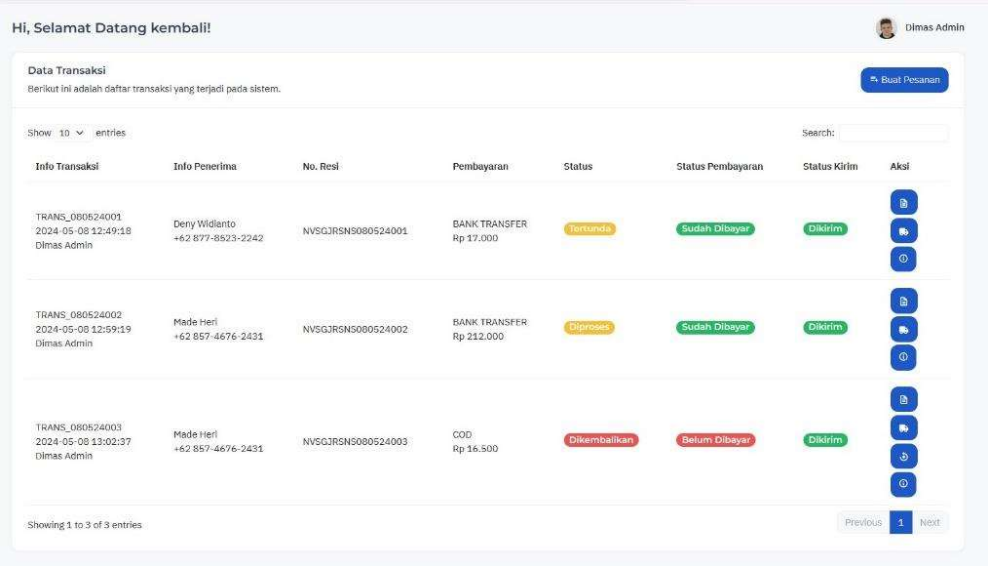
Gambar 4. 1 Implementasi Database

4.1.3 Implementasi Program

Berikut ini adalah hasil implementasi program :

1. Melihat Daftar Transaksi

Fitur ini digunakan admin untuk melihat daftar transaksi yang telah dibuat untuk menambahkan data retur berdasarkan id transaksi. Pada halaman transaksi akan ditampilkan setiap data transaksi.



Gambar 4. 2 Implementasi Fitur Melihat Daftar Transaksi

```
public function index(Request $request)
  {
    if (empty(session('user'))) {
      return redirect('');
    }
    $data['title'] = 'Transaksi';

    $data['transactions'] = $this->queryGetTrans();

    if($request->ajax()){
      $transaksi = Transaction::with('client', 'details')->firstWhere('id_trans', $request->id_trans);
      $detailTrans = TransactionDetail::with('produk')->where('id_trans', $transaksi->id_trans)->get();
      return response()->json([
        'status' => true,
        'message' => 'Data Transaksi',
        'data' => $transaksi,
        'detailTrans' => $detailTrans
      ]);
    }

    $month = date('n');  // Mendapatkan angka bulan (tanpa nol di depan)
    $year = date('Y');  // Mendapatkan angka tahun (4 digit)

    $res    = $this->queryGetTransaction($month, $year);
    $total_days = cal_days_in_month(CAL_GREGORIAN, $month, $year);
    $dataTrans = array();
    for ($i = 1; $i <= $total_days; $i++) {
      $res = $this->queryGetTransaction($month, $year);
      $dt = sprintf('%02d', $i);
      $dt = $year.'-'.date('m').'-'.$dt;

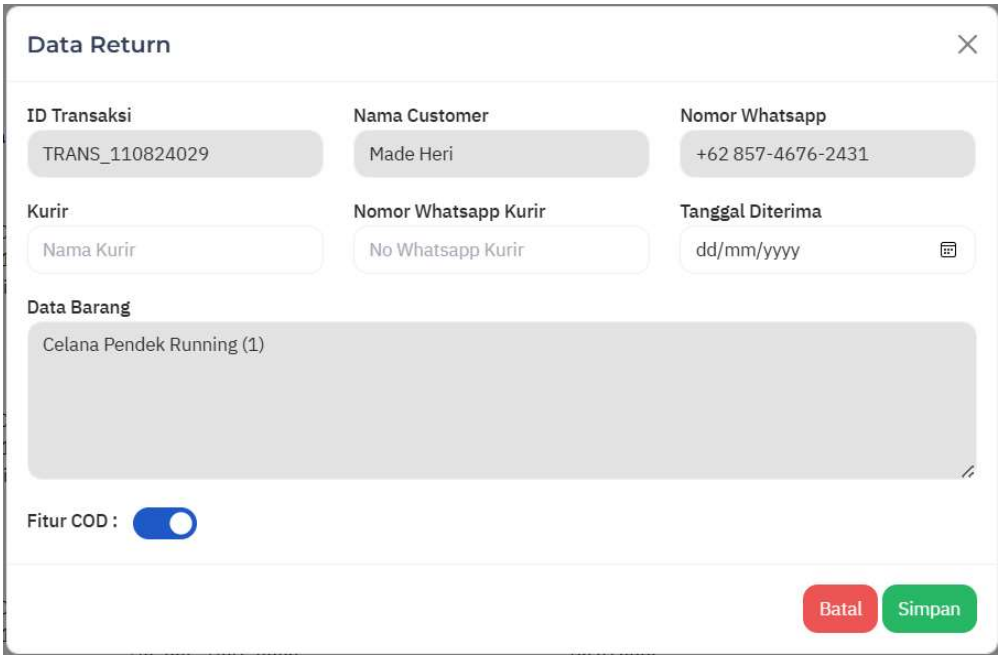
      array_push($dataTrans, $res[$dt]);
    }
    $data['trans'] = $dataTrans;

    // dd($data);
    return
      view('template_main.header', $data) .
      view('template_main.sidebar', $data) .
      view('template_main.main_page.transaction', $data) .
      view('template_main.footer');
  }
```

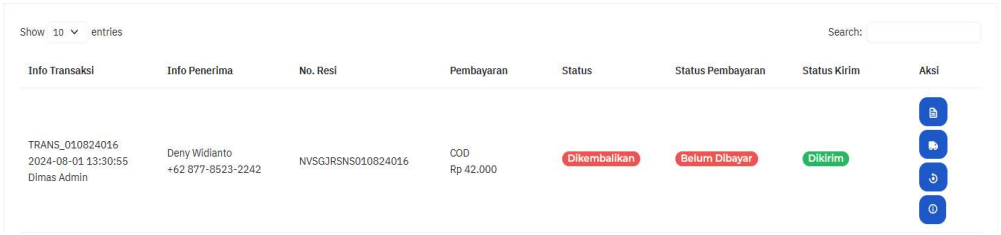
Segmen Program 4. 1 Transaction Controller

2. Tambah Data Retur

Fitur ini digunakan untuk menambahkan data retur baru. Klien akan menghubungi admin melalui *whatsapp* untuk *request* pengembalian barang. Jika mendapat persetujuan, admin akan mengonfirmasi kepada klien terkait pengembalian barang maupun kirim ulang pesanan. Saat barang sudah sampai di kantor admin akan mengisi data kurir, dan tanggal barang masuk ke gudang. Menambahkan data retur akan mengubah status pada transaksi menjadi "dikembalikan".



Gambar 4. 3 Implementasi Fitur Tambah Data Retur



Gambar 4. 4 Output Setelah Tambah Retur

```
function saveReturn(Request $request){
    try {
        $return = DB::table('return')->where('id_trans', $request->input('id_transaksi'))->first();
        if($return != null){
            return response()->json([
                'status' => false,
                'message' => "Data sudah ada di table return"
            ]);
        }else{
            $data = [
                'id_return' => 'RTRN-' . randomString(6),
                'id_trans' => $request->input('id_transaksi'),
                'nama_kurir' => $request->input('nama_kurir'),
                'no_wa_kurir' => $request->input('no_wa_kurir'),
                'tgl_diterima' => $request->input('tgl_diterima'),
                'tgl_return' => now()->format('Y-m-d'),
                'waktu_return' => now()->format('H:i:s'),
                'status_return' => 0
            ];

            DB::table('return')->insert($data);

            $klien = DB::table('md_klien')->where('id_klien', $request->input('id_klien'))->update([
                'status_COD' => $request->input('is_active') == 'on' ? 1 : 0,
            ]);

            $trans = Transaction::where('id_trans', $request->input('id_transaksi'))->update([
                'status' => 4,
            ]);

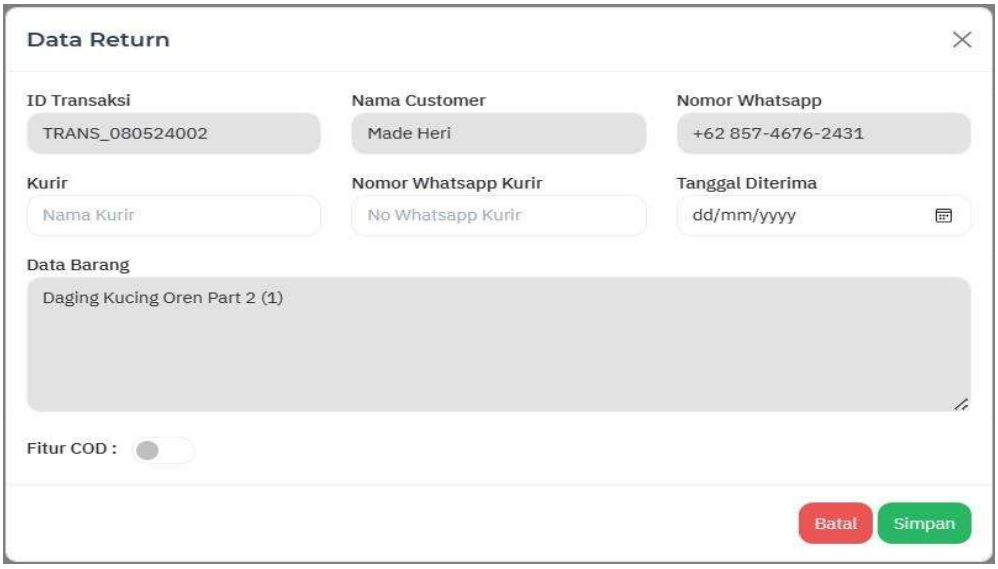
            if($request->input('is_active') != 'on'){
                $phone = str_replace('+62', '62', $request->input('no_whatsapp'));
                $phone = str_replace('-', '', $phone);
                $phone = str_replace(' ', '', $phone);
                $message = "Pesanan COD dengan ID Transaksi " . $request->input('id_transaksi') . " tidak terbayar.
                Anda tidak dapat menggunakan metode pembayaran COD untuk transaksi selanjutnya.";
                $wa = Whatsapp::sendMessage($phone, $message);
            }

            return response()->json([
                'status' => true,
                'message' => "Merubah transaksi ke status dikembalikan"
            ]);
        }
    } catch (\Throwable $th) {
        Log::info($th->getMessage());
        DB::table('return')->where('id_trans', $request->input('id_transaksi'))->delete();
        return response()->json([
            'status' => false,
            'message' => "Gagal merubah transaksi ke status dikembalikan"
        ]);
    }
}
```

Segmen Program 4. 2 Return Controller

3. Menonaktifkan Fitur COD

Fitur ini memungkinkan admin untuk menonaktifkan fitur pembayaran COD apabila customer tidak melakukan pembayaran terhadap pesanan yang sudah dikirim. Jika fitur cod dimatikan, sistem akan otomatis mengirim notifikasi kepada klien.



Gambar 4. 5 Implementasi Fitur Menonaktifkan Fitur COD

```
function saveReturn(Request $request){
    try {
        $return = DB::table('return')->where('id_trans', $request->input('id_transaksi'))->first();
        if($return != null){
            return response()->json([
                'status' => false,
                'message' => "Data sudah ada di table return"
            ]);
        }else{
            $data = [
                'id_return' => 'RTRN-' . randomString(6),
                'id_trans' => $request->input('id_transaksi'),
                'nama_kurir' => $request->input('nama_kurir'),
                'no_wa_kurir' => $request->input('no_wa_kurir'),
                'tgl_diterima' => $request->input('tgl_diterima'),
                'tgl_return' => now()->format('Y-m-d'),
                'waktu_return' => now()->format('H:i:s'),
                'status_return' => 0
            ];

            DB::table('return')->insert($data);

            $klien = DB::table('md_klien')->where('id_klien', $request->input('id_klien'))->update([
                'status_COD' => $request->input('is_active') == 'on' ? 1 : 0,
            ]);

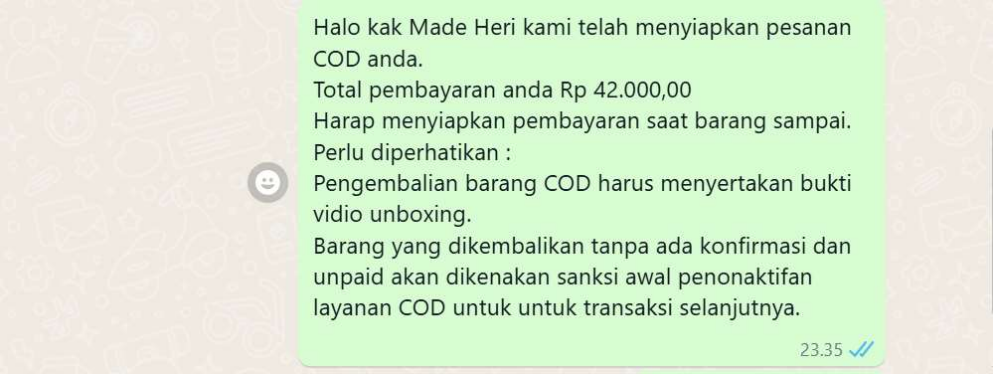
            $trans = Transaction::where('id_trans', $request->input('id_transaksi'))->update([
                'status' => 4,
            ]);

            if($request->input('is_active') != 'on'){
                $phone = str_replace('+62', '62', $request->input('no_whatsapp'));
                $phone = str_replace('-', '', $phone);
                $phone = str_replace(' ', '', $phone);
                $message = "Pesanan COD dengan ID Transaksi " . $request->input('id_transaksi') . " tidak terbayar.
                Anda tidak dapat menggunakan metode pembayaran COD untuk transaksi selanjutnya.";
                $wa = Whatsapp::sendMessage($phone, $message);
            }

            return response()->json([
                'status' => true,
                'message' => "Merubah transaksi ke status dikembalikan"
            ]);
        }
    } catch (\Throwable $th) {
        Log::info($th->getMessage());
        DB::table('return')->where('id_trans', $request->input('id_transaksi'))->delete();
        return response()->json([
            'status' => false,
            'message' => "Gagal merubah transaksi ke status dikembalikan"
        ]);
    }
}
```

Segmen Program 4. 3 Return Controller

Mekanisme COD dikirim melalui *whatsapp gateway* berdasarkan parameter nomor wa bersamaan dengan input data pesanan. Saat *submit* data pesanan sistem akan mengecek metode pembayaran yang digunakan. Apabila metode pembayaran yang digunakan adalah COD, sistem akan mengirim notifikasi terkait mekanisme COD.



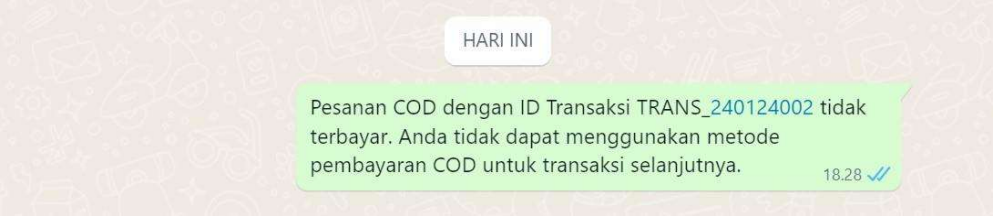
Gambar 4. 6 Output Notifikasi Mekanisme COD

```
if ($req->input('pay_method') == 'COD') {
            Whatsapp::sendMessage($klien->no_wa, [
                'Halo kak [name] kami telah menyiapkan pesanan COD anda.',
                'Total pembayaran anda [harga]',
                'Harap menyiapkan pembayaran saat barang sampai.',
                'Perlu diperhatikan :',
                'Pengembalian barang COD harus menyertakan bukti video unboxing.',
                'Barang yang dikembalikan tanpa ada konfirmasi dan unpaid akan dikenakan sanksi awal
                 penonaktifan layanan COD untuk untuk transaksi selanjutnya.'
            ], [
                'name' => $klien->nama_klien,
                'harga' => Number::currency($data_trans['total_bayar'], in: 'IDR', locale: 'id_ID'),
            ]);
        }
        $sendNotif = true;
                return redirect('order')->with('resp_sts', "success")->with('resp_msg', "Berhasil menambahkan pesanan");
    } catch (HttpException $e) {
        Log::info($e->getMessage());
        return redirect('order')->with('resp_sts', "error")->with('resp_msg', 'Gagal menambahkan pesanan,
        error : ' . $e->getMessage());
    }
```

Segmen Program 4. 4 Return Controller

4. Kirim Notifikasi Metode Pembayaran COD Dinonaktifkan

Notifikasi akan dikirimkan secara otomatis dengan parameter nomor wa customer apabila fitur COD dinonaktifkan sehingga customer dapat mengetahui dan melakukan konfirmasi.



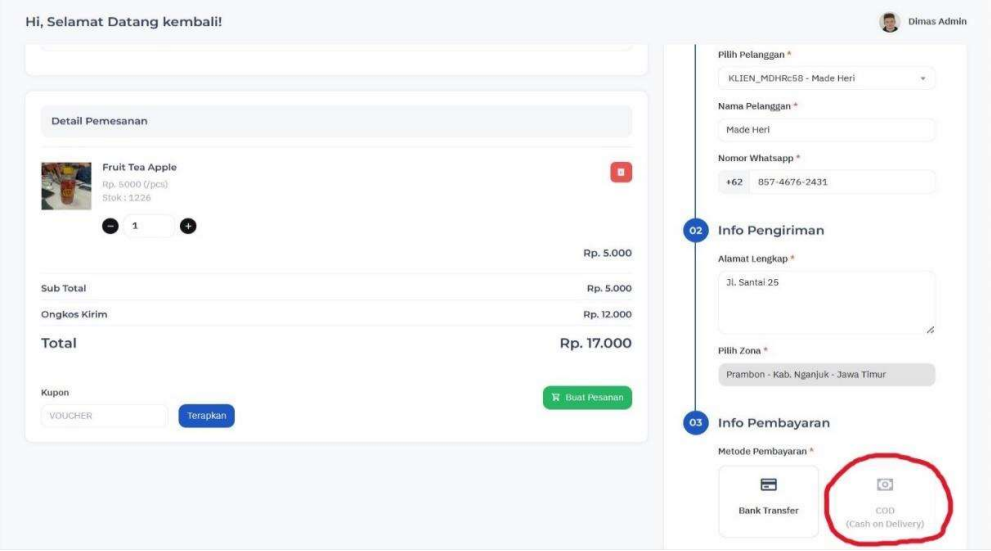
Gambar 4. 7 Implementasi Fitur Kirim Notifikasi

```
        if($request->input('is_active') != 'on'){
            $phone = str_replace('+62', '62', $request->input('no_whatsapp'));
            $phone = str_replace('-', '', $phone);
            $phone = str_replace(' ', '', $phone);
            $message = "Pesanan COD dengan ID Transaksi " . $request->input('id_transaksi') . " tidak terbayar.
            Anda tidak dapat menggunakan metode pembayaran COD untuk transaksi selanjutnya.";
            $wa = Whatsapp::sendMessage($phone, $message);
        }

        return response()->json([
            'status' => true,
            'message' => "Merubah transaksi ke status dikembalikan"
        ]);
    }
}
```

Segmen Program 4. 5 Return Controller

Setelah fitur COD dinonaktifkan dan berhasil mengirimkan pesan, pada transaksi selanjutnya klien tidak bisa menggunakan metode pembayaran COD.




```
<script>
  $('#klien').on('change', function(){
    let val = $(this).val()
    if(val == 'klien baru'){
      $('#nama').val('')
      $('#no_wa').val('')
      $('#alamat').html('')
      $('#form-zone-txt').addClass('d-none')
      $('#form-zone').removeClass('d-none')
    }else{
      $.ajax({
        url: location.href + '?id_klien=' + $(this).val(),
        success: function(res){
          $('#form-zone').addClass('d-none')
          $('#form-zone-txt').removeClass('d-none')
          $('#nama').val(res.nama_klien)
          $('#no_wa').val((res.no_wa).replace('+62 ', ''))
          $('#alamat').html(res.alamat)
          $('#zone_txt').val(`${res.kecamatan} - ${res.kota} - ${res.provinsi}`)
          if (res.status_COD == '0') {
            $('#field-cod').addClass('disabledbutton')
          }else{
            $('#field-cod').removeClass('disabledbutton')
          }

          $.ajax({
            url: location.origin + '/zone/search?term=' + res.kecamatan,
            success: function(res){
              $('input[name="l1_code"]').val(res[0]['l1_code'])
              $('input[name="l2_code"]').val(res[0]['l2_code'])/

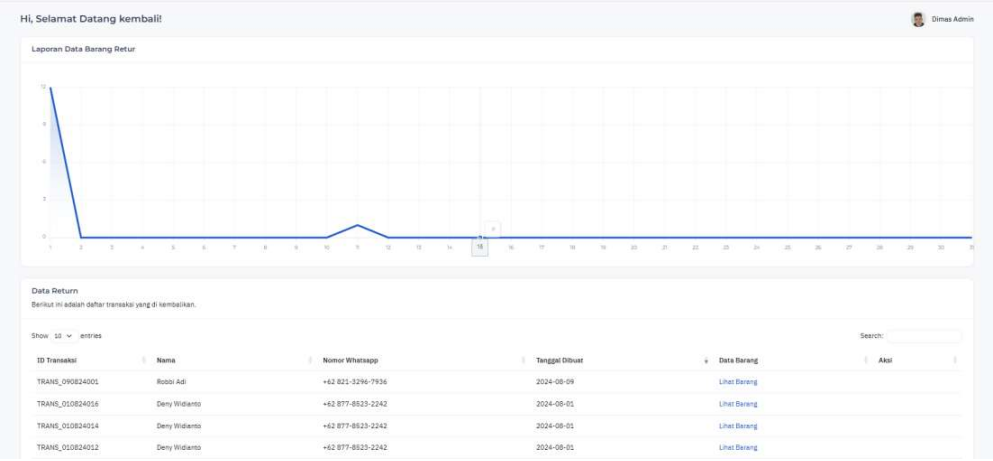
              $('input[name="id_zona"]').val(res[0]['id_zone'])
              $('input[name="kecamatan"]').val(res[0]['kecamatan'])
              $('input[name="kota"]').val(res[0]['kota'])
              $('input[name="provinsi"]').val(res[0]['provinsi'])

              HitungOngkir()
            }
          })
        }
      })
    }
  })
</script>
```

Segmen Program 4. 6 Order Controller

5. Melihat Daftar Retur

Fitur ini digunakan untuk melihat daftar retur yang sudah direcord. di halaman daftar retur ini gudang dapat melakukan pengecekan terhadap kondisi barang retur dan mencetak laporan. Pada halaman retur ditampilkan grafik transaksi yang diretur perhari.



Gambar 4. 8 Implementasi Fitur Melihat Daftar Retur

```
public function index(Request $request)
    {
        if (empty(session('user'))) {
            return redirect('');
        }
        if($request->ajax()){
            $transaksi = Transaction::with('client', 'details')->firstWhere('id_trans', $request->id_trans);
            $detailTrans = TransactionDetail::with('produk')->where('id_trans', $transaksi->id_trans)->get();
            $return = DB::table('return')->where('id_trans', $request->id_trans)->first();
            return response()->json([
                'status' => true,
                'message' => 'Data Transaksi',
                'data' => $transaksi,
                'detailTrans' => $detailTrans,
                'return' => $return
            ]);
        }
        $data['title'] = 'Return';

        // $data['transactions'] = $this->queryGetTransRefund();
        $data['transactions'] = TransReturn::with('transaction.details', 'transaction.client')->
        orderBy('waktu_return', 'DESC')->get();

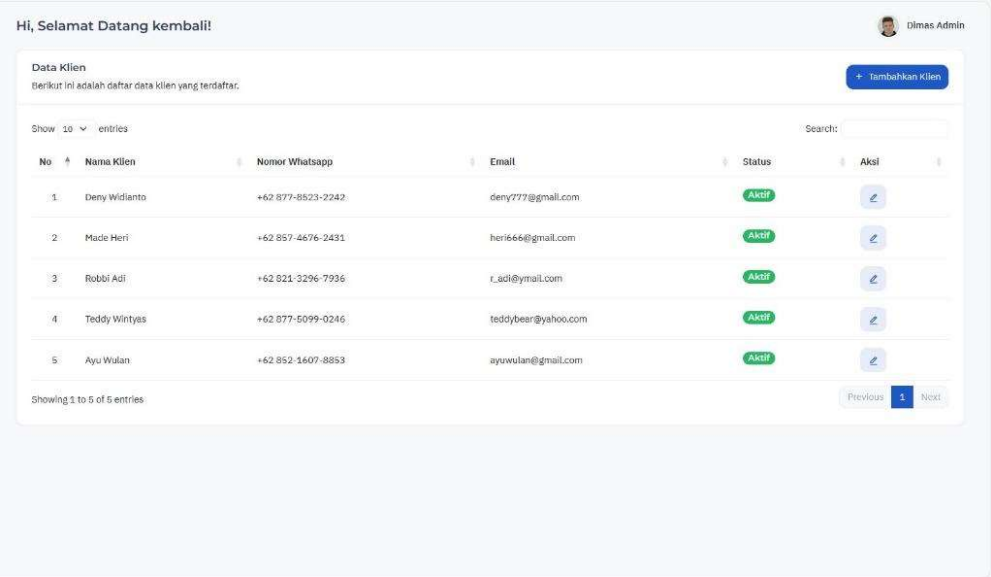
        $month = date('n');  // Mendapatkan angka bulan (tanpa nol di depan)
        $year = date('Y');  // Mendapatkan angka tahun (4 digit)
        $res    = $this->queryGetReturn($month, $year);
        $total_days = cal_days_in_month(CAL_GREGORIAN, $month, $year);
        $dataReturn = array();
        for ($i = 1; $i <= $total_days; $i++) {
            $res = $this->queryGetReturn($month, $year);
            $dt = sprintf('%02d', $i);
            $dt = $year.'-'.date('m').'-'.$dt;
            array_push($dataReturn, $res[$dt]);
        }
        $data['return'] = $dataReturn;

        return
            view('template_main.header', $data) .
            view('template_main.sidebar', $data) .
            view('template_main.side_page_gudang.return', $data) .
            view('template_main.footer');
    }
```

Segmen Program 4. 7 Return Controller

6. Melihat Daftar Klien

Fitur ini berfungsi untuk menampilkan semua data klien yang terdaftar. Pada halaman klien admin dapat menambahkan, mengubah dan menghapus data klien.



Gambar 4. 9 Implementasi Fitur Melihat Daftar Klien

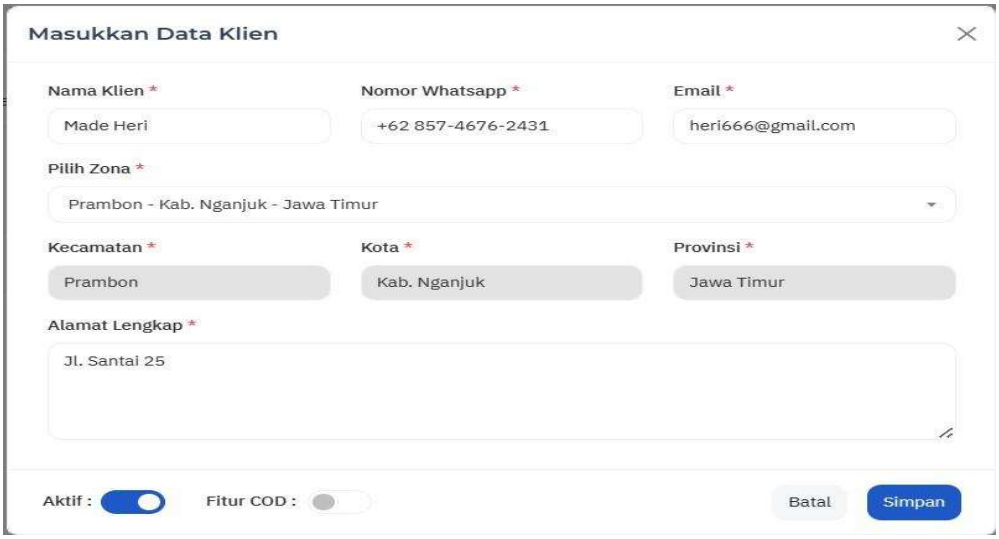
```
public function index()
    {
        if (empty(session('user'))) {
            return redirect('');
        }
        $data['title'] = 'Klien';
        $data['klien'] = DB::select('
            SELECT
                mk.*
            FROM
                md_klien mk
        ');

        return
            view('template_main.header', $data) .
            view('template_main.sidebar', $data) .
            view('template_main.side_page_admin.klien', $data) .
            view('template_main.footer');
    }
```

Segmen Program 4. 8 Klien Controller

7. Setting Metode Pembayaran

Fitur ini memungkinkan admin mengaktifkan kembali fitur pembayaran COD. Pada *database* klien terdapat status yang terhubung dengan *button slider* fitur COD. Status = 0 berarti fitur COD mati, status = 1 berarti fitur COD aktif. Saat menambahkan pesanan baru, admin menginputkan data klien, kemudian sistem akan mengecek status COD pada data klien. Jika status dalam *database* 1 maka *button* metode pembayaran COD akan aktif, begitu pula sebaliknya.



Gambar 4. 10 Implementasi Fitur Setting Metode Pembayaran

```
public function edit(Request $req)
  {
    $klien = DB::selectOne('
      SELECT
        mk.*
      FROM
        md_klien mk
      WHERE
        mk.id_klien = "' . $req->input('id_klien') . '"
    ');
    try {
      $data = [
        'nama_klien' => $req->input('nama'),
        'no_wa' => $req->input('no_wa'),
        'email' => $req->input('email'),
        'id_zone' => $req->input('id_zona'),
        'kecamatan' => $req->input('kecamatan'),
        'kota' => $req->input('kota'),
```

```
            'provinsi' => $req->input('provinsi'),
                'alamat' => $req->input('alamat'),
                'is_active' => ($req->input('is_active') == 'on') ? 1 : 0,
                'status_cod' => ($req->input('status_cod') == 'on') ? 1 : 0
            ];

            if (!$req->input('is_active')) {
                if ($klien->is_active == 1) {
                    $data['log_disabled'] = date('Y-m-d H:i:s');
                }
            } else {
                $data['log_disabled'] = NULL;
            }

            DB::table('md_klien')->where(['id_klien' => $req->input('id_klien')])->update($data);

            return redirect($req->input('redirect_to'))->with('resp_sts', 'success')->
            with('resp_msg', 'Berhail megubah data klien.');

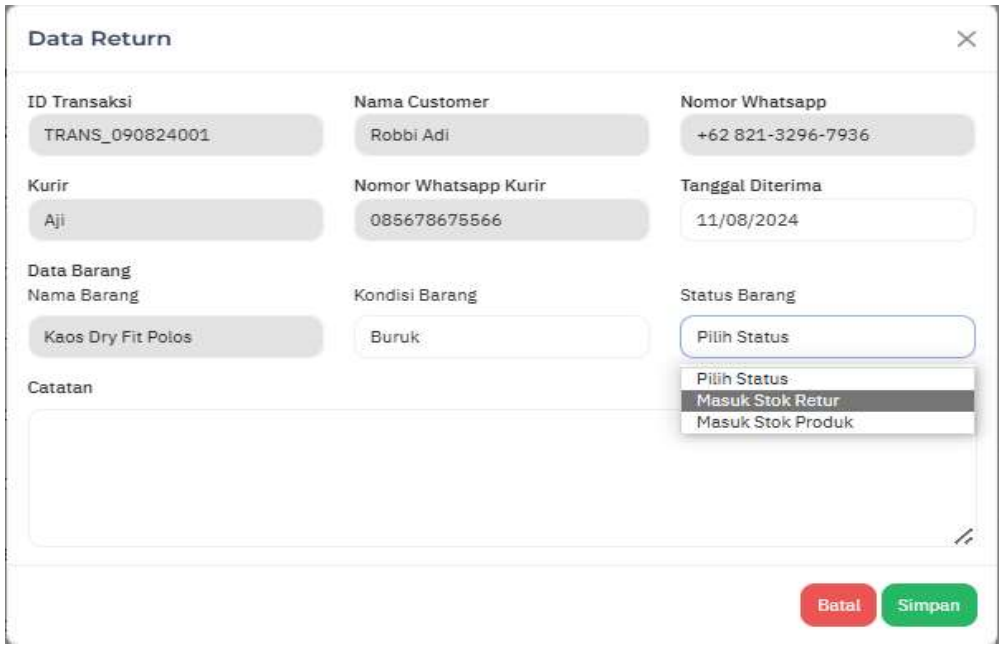
        } catch (HttpException $e) {
            return redirect($req->input('redirect_to'))->with('resp_sts', 'error')->
            with('resp_msg', 'Gagal mengubah data klien, error : ' . $e->getMessage() );

        }
    }
```

Segmen Program 4. 9 Klien Controller

8. Cek Kondisi Barang

Fitur ini memungkinkan gudang untuk melakukan pencatatan kondisi barang retur sesuai dengan kondisi fisik yang ada. Dalam sistem ditentukan jika kondisi barang buruk maka harus masuk stok retur, begitu pula sebaliknya jika kondisi barang baik akan masuk stok produk. Jika input tidak sesuai maka akan menampilkan pesan error. Jika berhasil akan mengupdate data pada database agar stok fisik dan stok dalam database tidak terjadi selisih.



Gambar 4. 11 Implementasi Fitur Cek Kondisi Barang

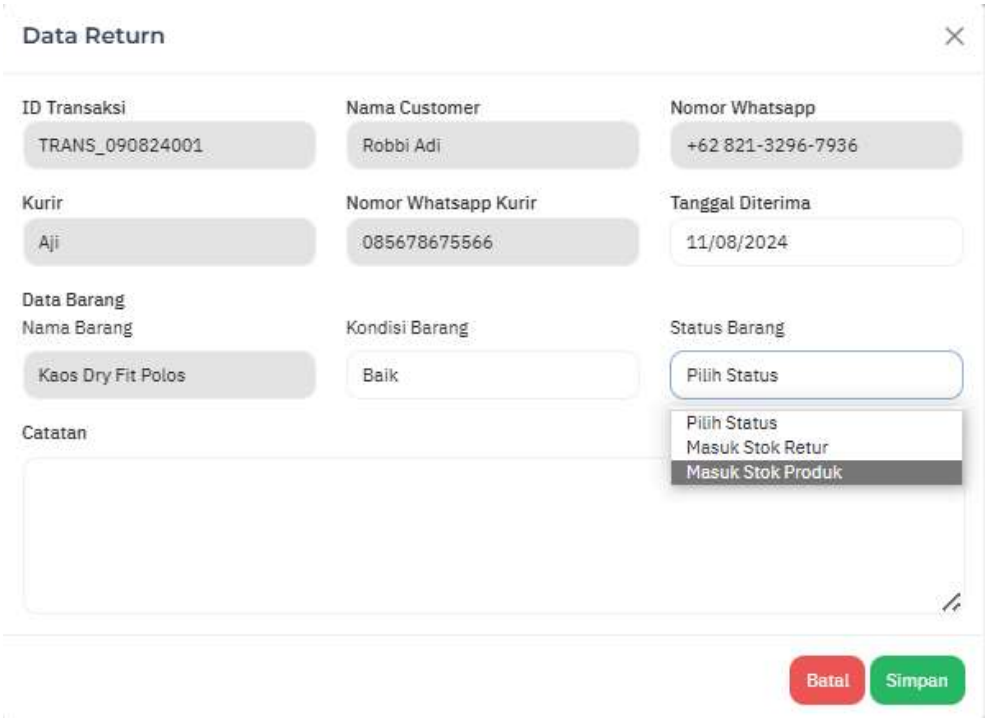
```
function editReturn(Request $request){
    try {
        for ($i=0; $i < count($request->idproduk); $i++) {
            $detTrans = TransactionDetail::where('id_trans', $request->idtrans[$i])->where('id_produk', $request->idproduk[$i])->first();
            if($detTrans->status_produk != $request->status_produk[$i] && $detTrans->kondisi_produk != $request->kondisi_produk[$i]){
                if(($request->kondisi_produk[$i] == 'Baik') && ($request->status_produk[$i] == 'Masuk Stok Produk')){
                    $produk = Produk::where('id_produk', $request->idproduk[$i])->first();
                    $produk->stok = (int)$produk->stok + (int)$detTrans->qty;
                    // if($produk->stok_retur != 0 || $produk->stok_retur != null){
                    //     $produk->stok_retur = (int)$produk->stok_retur - (int)$detTrans->qty;
                    // }
                    $produk->save();
                }elseif(($request->kondisi_produk[$i] == 'Buruk') && ($request->status_produk[$i] == 'Masuk Stok Retur')){
                    $produk = Produk::where('id_produk', $request->idproduk[$i])->first();
                    $produk->stok_retur = (int)$produk->stok_retur + (int)$detTrans->qty;
                    if($produk->stok != 0 || $produk->stok != null){
                        $produk->stok = (int)$produk->stok - (int)$detTrans->qty;
                    }
                    $produk->save();
                }
```

```
$transDetail = TransactionDetail::where('id_trans', $request->idtrans[$i])->where('id_produk', $request->idproduk[$i])->update([
                    'status_produk' => $request->status_produk[$i],
                    'kondisi_produk' => $request->kondisi_produk[$i],
                ]);
            }
        }
        $return  = TransReturn::where('id_return', $request->id_return)->update([
            'catatan' => $request->catatan
        ]);
        return response()->json([
            'status' => true,
            'message' => "Berhasil mengupdate stok produk"
        ]);
    } catch (\Throwable $th) {
        Log::info($th->getMessage());
        return response()->json([
            'status' => false,
            'message' => "Gagal mengupdate stok produk"
        ]);
    }
}
```

Segmen Program 4. 10 Return Controller

9. Edit Kondisi Barang

Fitur ini memungkinkan gudang untuk melakukan pengubahan kondisi barang yang sudah tercatat. Kesalahan input yang disebabkan oleh human error bisa saja terjadi. Untuk itu perlu ada fitur edit dan kemudian mengupdate kembali data pada database agar stok fisik dan stok dalam database tidak terjadi selisih.



Gambar 4. 12 Implementasi Fitur Edit Kondisi Barang

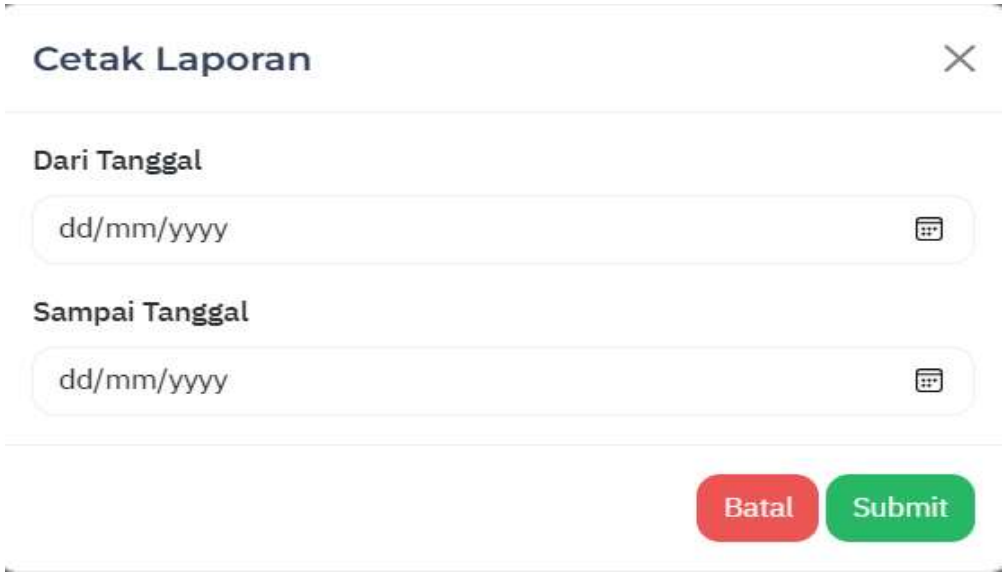
```
function editReturn(Request $request){
    try {
      for ($i=0; $i < count($request->idproduk); $i++) {
        $detTrans = TransactionDetail::where('id_trans', $request->idtrans[$i])->where('id_produk', $request->idproduk[$i])->first();
        if($detTrans->status_produk != $request->status_produk[$i] && $detTrans->kondisi_produk != $request->kondisi_produk[$i]){
          if(($request->kondisi_produk[$i] == 'Baik') && ($request->status_produk[$i] == 'Masuk Stok Produk')){
            $produk = Produk::where('id_produk', $request->idproduk[$i])->first();
            $produk->stok = (int)$produk->stok + (int)$detTrans->qty;
            // if($produk->stok_retur != 0 || $produk->stok_retur != null){
            //     $produk->stok_retur = (int)$produk->stok_retur - (int)$detTrans->qty;
            // }
            $produk->save();
          }elseif(($request->kondisi_produk[$i] == 'Buruk') && ($request->status_produk[$i] == 'Masuk Stok Retur')){
            $produk = Produk::where('id_produk', $request->idproduk[$i])->first();
            $produk->stok_retur = (int)$produk->stok_retur + (int)$detTrans->qty;
            if($produk->stok != 0 || $produk->stok != null){
              $produk->stok = (int)$produk->stok - (int)$detTrans->qty;
            }
            $produk->save();
          }
```

```
$transDetail = TransactionDetail::where('id_trans', $request->idtrans[$i])->where('id_produk', $request->idproduk[$i])->update([
                    'status_produk' => $request->status_produk[$i],
                    'kondisi_produk' => $request->kondisi_produk[$i],
                ]);
            }
        }
        $return  = TransReturn::where('id_return', $request->id_return)->update([
            'catatan' => $request->catatan
        ]);
        return response()->json([
            'status' => true,
            'message' => "Berhasil mengupdate stok produk"
        ]);
    } catch (\Throwable $th) {
        Log::info($th->getMessage());
        return response()->json([
            'status' => false,
            'message' => "Gagal mengupdate stok produk"
        ]);
    }
}
```

Segmen Program 4. 11 Return Controller

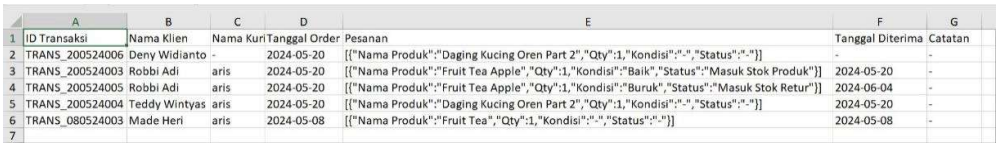
10. Cetak Rekap Pengelolaan Retur

Fitur ini memungkinkan gudang untuk mencetak laporan retur. Periode laporan yang ingin dicetak bisa diatur berdasarkan tanggal yang diinginkan. Jika disubmit ouput yang dihasilkan akan berupa format excel.



Gambar 4. 13 Implementasi Fitur Cetak Rekap Pengelolaan Retur

Output laporan akan berupa format excel yang akan menampilkan seluruh detail transaksi yang sudah diretur. Laporan berfungsi untuk cek fisik ulang pada gudang.



	A	B	C	D	E	F	G
1	ID Transaksi	Nama Klien	Nama Kuri	Tanggal Order	Pesanan	Tanggal Diterima	Catatan
2	TRANS_200524006	Deny Widianto	-	2024-05-20	[{"Nama Produk":"Daging Kucing Oren Part 2","Qty":1,"Kondisi":"-","Status":"-"}]	-	-
3	TRANS_200524003	Robbi Adi	aris	2024-05-20	[{"Nama Produk":"Fruit Tea Apple","Qty":1,"Kondisi":"Baik","Status":"Masuk Stok Produk"}]	2024-05-20	-
4	TRANS_200524005	Robbi Adi	aris	2024-05-20	[{"Nama Produk":"Fruit Tea Apple","Qty":1,"Kondisi":"Buruk","Status":"Masuk Stok Retur"}]	2024-06-04	-
5	TRANS_200524004	Teddy Wintyas	aris	2024-05-20	[{"Nama Produk":"Daging Kucing Oren Part 2","Qty":1,"Kondisi":"-","Status":"-"}]	2024-05-20	-
6	TRANS_080524003	Made Heri	aris	2024-05-08	[{"Nama Produk":"Fruit Tea","Qty":1,"Kondisi":"-","Status":"-"}]	2024-05-08	-
7							

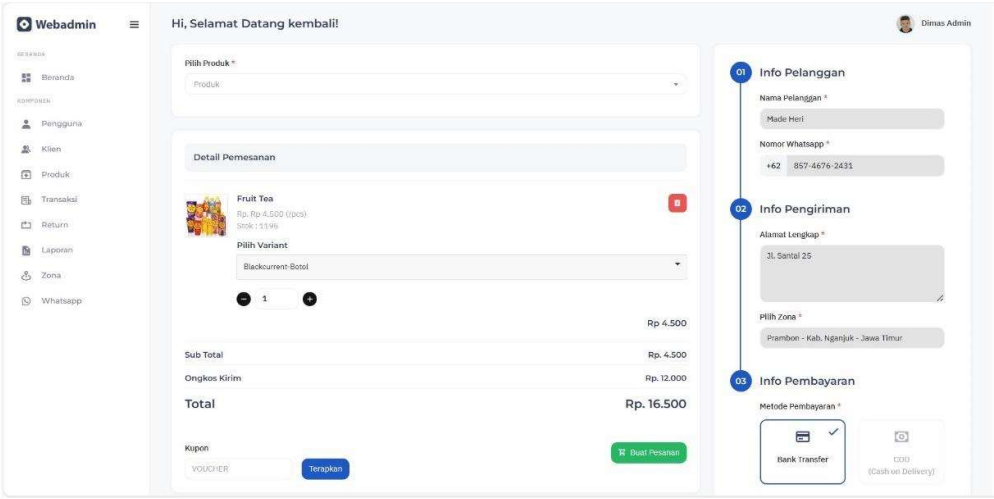
Gambar 4. 14 Output Laporan

```php
public function cetakLaporan(Request $request)
 {
    $from = date_format(date_create($request->dari_tgl), 'Y-m-d');
    $to   = date_format(date_create($request->sampai_tgl), 'Y-m-d');
    $data = [];
    $return = TransReturn::with('details', 'transaction', 'details.produk', 'transaction.client')->whereBetween('tgl_return', [$from, $to])->get();
    foreach ($return as $value) {
       $detTrans = TransactionDetail::with('produk')->where('id_trans', $value->transaction->id_trans)->get();
       $details = array();
       foreach ($detTrans as $key) {
          $pesanan = (object)[
             // 'ID Produk' => $key->id_produk ?? '-',
             'Nama Produk' => $key->produk->nama_produk ?? '-',
             'Qty' => $key->qty ?? '-',
             'Kondisi' => $key->kondisi_produk ?? '-',
             'Status' => $key->status_produk ?? '-',
          ];
          $details[] = $pesanan;
       }
       $data[] = [
          'ID Transaksi' => $value->transaction->id_trans ?? '-',
          'Nama Klien' => $value->transaction->client->nama_klien ?? '-',
          'Nama Kurir' => $value->nama_kurir ?? '-',
          'Tanggal Order' => $value->transaction->tgl_dibuat ?? '-',
          'Pesanan' => json_encode($details) ?? '-',
          'Tanggal Diterima' => $value->tgl_diterima ?? '-',
          'Catatan' => $value->catatan ?? '-'
       ];
    }
    return (new FastExcel($data))->download('Laporan_Return_' . Carbon::now()->format('d_m_Y') . '.xlsx');
 }
```

Segmen Program 4. 12 Return Controller

11. Kirim Ulang Pesanan

Fitur ini memungkinkan admin untuk menampilkan kembali data transaksi yang sudah diretur untuk melakukan kirim ulang pesanan. Detail transaksi seperti data klien dan detail pemesanan akan ditampilkan. Disini admin dapat mengubah, maupun menambahkan produk baru.



Gambar 4. 15 Implementasi Fitur Kirim Ulang Pesanan

```
function creditOrder(Request $req){
    $transaksi = Transaction::with('client', 'details', 'details.variant', 'details.produk')->firstWhere('id_trans', $req->id_trans);
    $data['trans'] = $transaksi;
    $data['title'] = 'Edit Transaksi';

    if($req->ajax()){
        $klien = Client::firstWhere('id_klien', $req->id_klien);
        return $klien;
    }

    return
        view('template_main.header', $data) .
        view('template_main.sidebar', $data) .
        view('template_main.side_page_admin.credit-order', $data) .
        view('template_main.footer');
  }
}
```

Segmen Program 4. 13 Order Controller

Pada proses pengiriman terdapat relasi dengan biaya kirim yang akan dihitung otomatis oleh API Ninja Van berdasarkan parameter alamat pada pengiriman dan

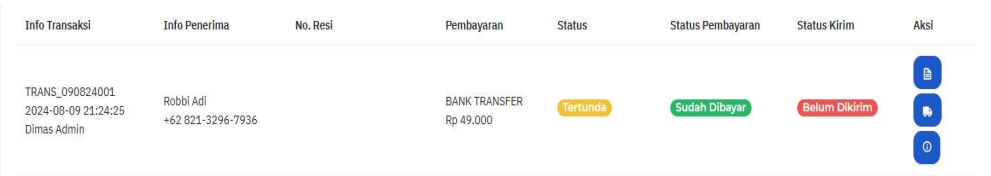
tujuan dan mengacu pada tabel harga ninja xpress. Berikut adalah syntax perhitungan biaya kirim :

```
{
 "weight": 0,
 "service_level": "Standard",
 "from": {
  "l1_tier_code": "string",
  "l2_tier_code": "string"
 },
 "to": {
  "l1_tier_code": "string",
  "l2_tier_code": "string"
 }
}


{
"data": {
"total_fee": 90000
}
}
```

Segmen Program 4. 14 Perhitungan Ongkir API Ninja Van

Untuk metode pembayaran transfer, aplikasi ini mengintegrasikan payment gateway midtrans dan whatsapp gateway untuk mengirimkan link pembayaran. Status pembayaran akan langsung update ketika klien berhasil melakukan pembayaran.



Gambar 4. 16 Update Status Otomatis

```
public function generatePayment(string $trxId, Request $request)
    {
        try {
            $trx = Transaction::firstWhere('id_trans', $trxId);

            if (empty($trx)) {
                throw new HttpException(404, 'Transaksi Tidak Ditemukan');
            }

            if ($trx->payment_url === null) {
                $trx->load(['details.produk', 'client']);
                $items = $trx->details->map(function ($item) {
                    return [
                        'id' => $item->id_produk,
                        'price' => $item->total_harga_produk / $item->qty,
                        'quantity' => $item->qty,
                        'name' => $item->produk->nama_produk ?? '-',
                        'merchant_name' => Payment::merchant_name(),
                        'ongkir' => 0
                    ];
                })->toArray();
                $items[] = [
                    'id' => 'ongkir',
                    'price' => $trx->harga_ongkir,
                    'quantity' => 1,
                    'name' => 'ongkir',
                    'merchant_name' => 'Ongkos Kirim',
                    'ongkir' => 0
                ];
                $ongkir = ['ongkir' => $trx->harga_ongkir];
                Log::info([$items, $ongkir]);
                $paymentUrl = Payment::snapCreateTrx($trxId, $trx->total_bayar, $items, $ongkir);
            } else {
                $paymentUrl = $trx->payment_url;
            }

            if ($trx->status_bayar === 0) {
                Whatsapp::sendMessage($trx->client->no_wa, [
                    'Halo kak [name] kami telah menerima pesanan kamu.',
                    'Silahkan lakukan pembayaran pesanmu dengan total [harga] menunggu dibayar',
                    'silahkan lakukan pembayaran di link berikut',
                    '[payment_url]'
                ], [
                    'name' => $trx->client->nama_klien,
                    'payment_url' => $paymentUrl,
                    'harga' => Number::currency($trx->total_bayar, in: 'IDR', locale: 'id_ID'),
                ]);
            }
            $trx->update(['status_bayar' => Transaction::UNPAID]);
            $sendNotif = true;
            return [
                'send_notif' => $sendNotif,
                'payment_url' => $paymentUrl,
                'message' => 'Pesan berhasil dikirim',
            ];
        } catch (\Throwable $th) {
            $sendNotif = false;
            return [
                'send_notif' => $sendNotif,
                'message' => 'Pesan gagal dikirim',
            ];
        }
    }
```

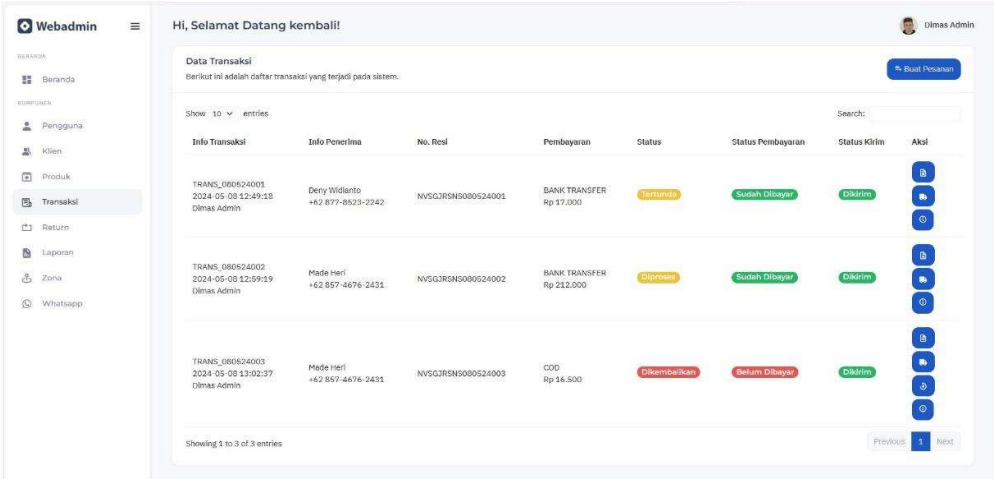
Segmen Program 4. 15 Update Status Pembayaran Midtrans

4.2 Uji Coba

Bagian ini menjelaskan tentang ujicoba yang dilakukan terhadap produk yang dihasilkan. Ujicoba *software* yang dilakukan peneliti pada penelitian ini adalah metode *black box*. Menurut (Tanza, 2022) dalam (Riky Andreansyah Bimantoro, 2024) *Black-Box Testing* merupakan bagian paling penting dalam proses pengembangan *software* yang melibatkan uji coba sistem. Uji coba berfokus pada spesifikasi fungsional *software* tanpa memeriksa desain dan kode program yang berhubungan dengan spesifikasi tersebut. Berikut merupakan hasil pengujian pada setiap tahapan prototype sesuai dengan role user :

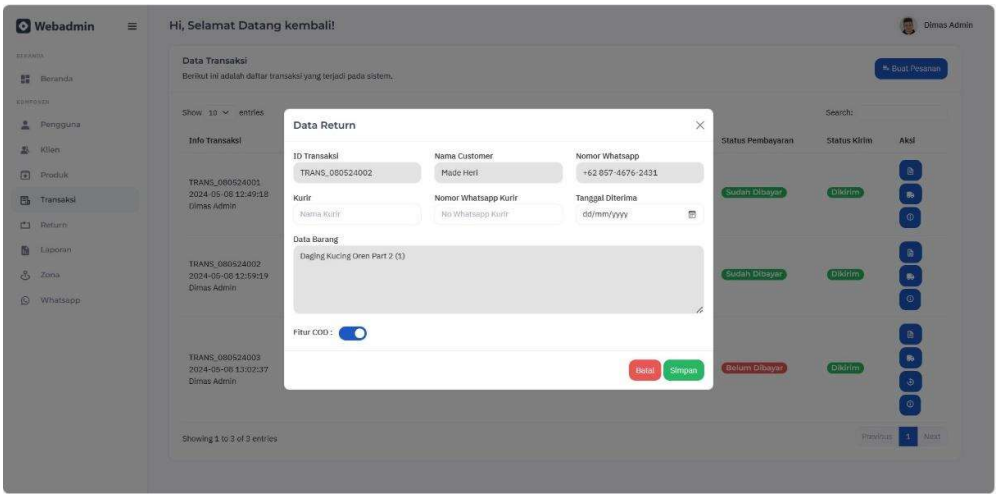
a. Admin

Pada tahapan pertama pengembangan aplikasi yaitu berfokus pada penambahan data barang retur. Role yang bertugas menambahkan data barang retur adalah role admin. Untuk menambahkan data barang retur admin harus membuka halaman daftar transaksi. Berikut adalah hasil output untuk pengujian melihat daftar transaksi.



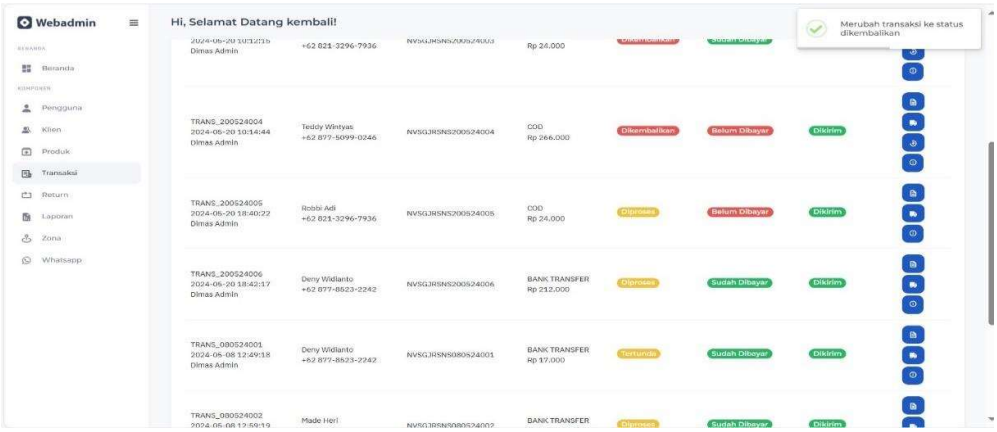
Gambar 4. 17 Prototype & Hasil Pengujian

Pada tahapan selanjutnya adalah menampilkan form data retur untuk menambahkan data retur. Berikut adalah output tampilan form data retur :



Gambar 4. 18 Prototype & Hasil Pengujian

Pada tahapan *quick design* yang diajukan kepada user, tampilan antar muka yang diusulkan berupa halaman penuh, namun tampilan antar muka yang disetujui menggunakan modal sehingga tidak perlu berpindah halaman. Jika mengklik button simpan maka data akan dimasukkan ke daftar retur, mengubah status transaksi menjadi dikembalikan dan menampilkan message.

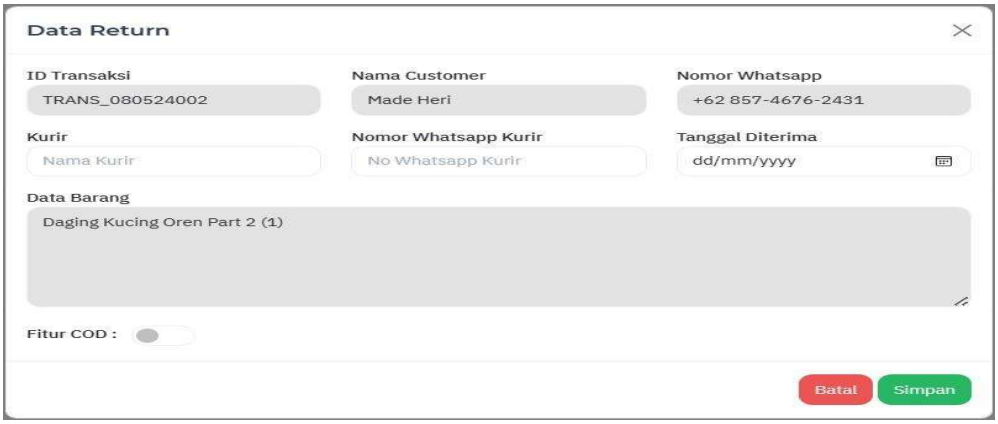


Gambar 4. 19 Prototype & Hasil Pengujian

Tabel 4. 1 Tabel Pengujian Prototype Tahap Pertama

Skenario	Pengujian	Hasil diharapkan	Hasil pengujian
Melihat daftar transaksi	Membuka halaman daftar transaksi	Berhasil menampilkan daftar transaksi	**Berhasil**
Menampilkan form data retur	Menampilkan form data retur dari id transaksi	Berhasil Menampilkan form data retur	**Berhasil**
Menambahkan data retur	Menambahkan data retur dari tabel transaksi	Berhasil menambahkan data retur, mengubah status transaksi dan menampilkan message	**Berhasil**

Pada tahapan kedua pengembangan aplikasi yaitu berfokus pada sistem notifikasi.



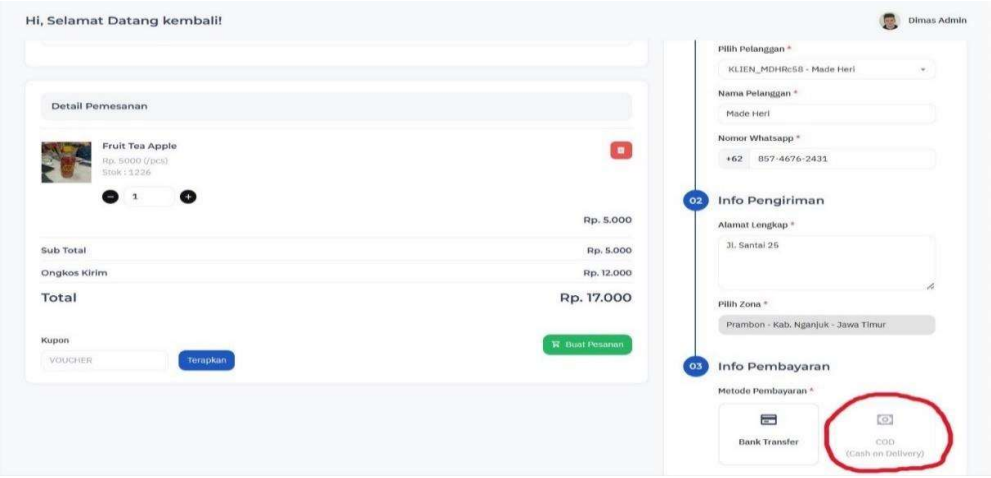
Gambar 4. 20 Prototype & Hasil Pengujian

Jika fitur pembayaran COD dinonaktifkan sistem akan mengirimkan pesan.



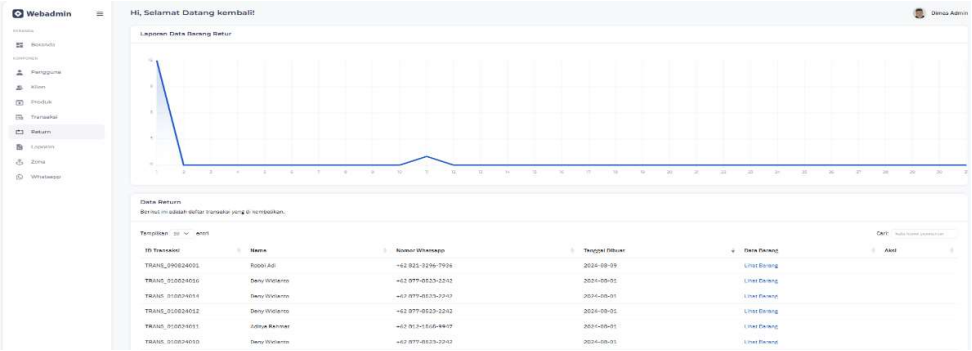
Gambar 4. 21 Prototype & Hasil Pengujian

Kemudian klien yang fitur pembayaran CODnya dinonaktifkan tidak dapat melakukan pembayaran COD pada transaksi baru.



Gambar 4. 22 Prototype & Hasil Pengujian

admin dapat melihat data retur pada halaman daftar retur.

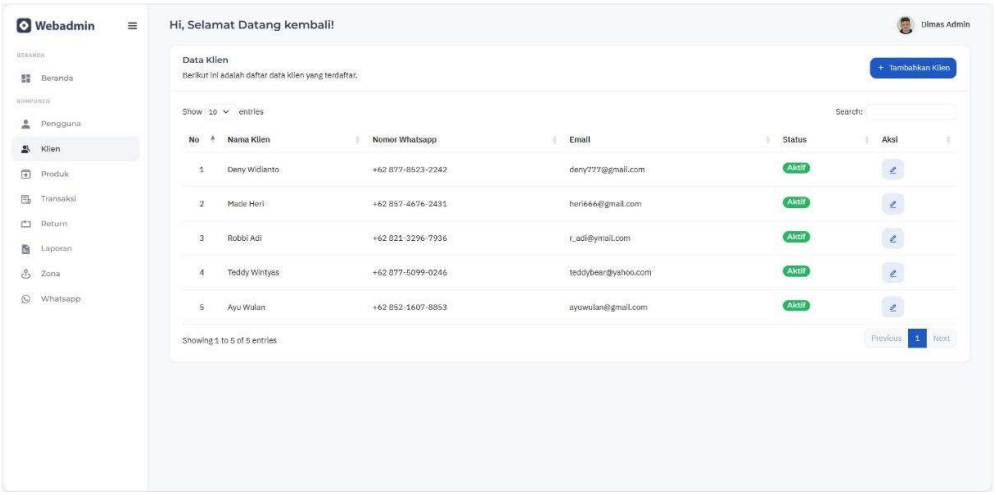


Gambar 4. 23 Prototype & Hasil Pengujian

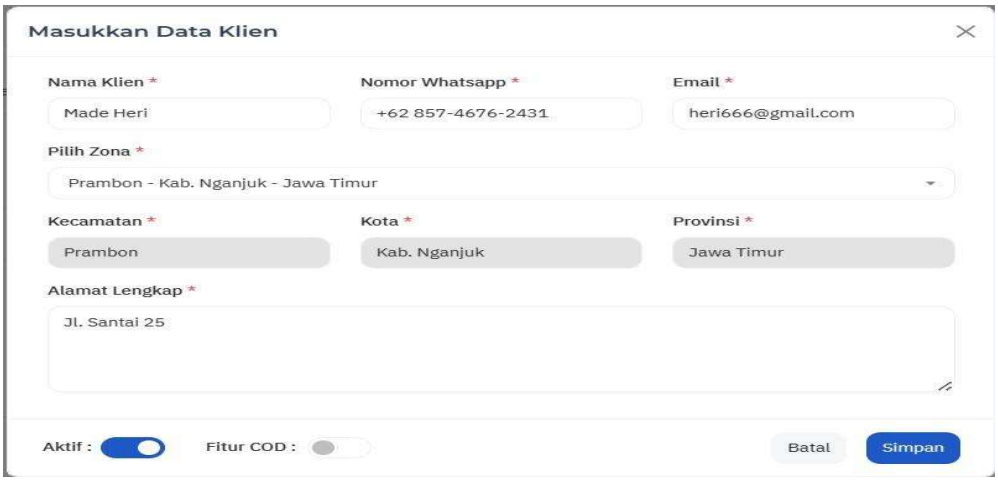
Tabel 4. 2 Tabel Pengujian Prototype Tahap Kedua

Skenario	Pengujian	Hasil diharapkan	Hasil pengujian
Menonaktifkan fitur pembayaran COD	Menonaktifkan fitur pembayaran COD saat mengisi form retur	Klien tidak dapat menggunakan fitur pembayaran COD pada transaksi selanjutnya.	**Berhasil**
Kirim Notifikasi Metode Pembayaran COD Dinonaktifkan	Menonaktifkan fitur pembayaran COD saat mengisi form retur	Mengirimkan pesan melalui wa kepada klien saat form retur berhasil di submit	**Berhasil**
Menambahkan transaksi baru	Menambahkan transaksi baru dengan id klien yang fitur pembayaran CODnya dinonaktifkan	Tidak dapat menggunakan fitur pembayaran COD	**Berhasil**
Melihat dafttar retur	Membuka halaman daftar retur	Berhasil menampilkan daftar retur	**Berhasil**

Pada tahapan ketiga pengembangan aplikasi yaitu berfokus pada fitur setting metode pembayaran. Fitur ini digunakan untuk mengaktifkan kembali fitur pembayaran COD klien yang dinonaktifkan.

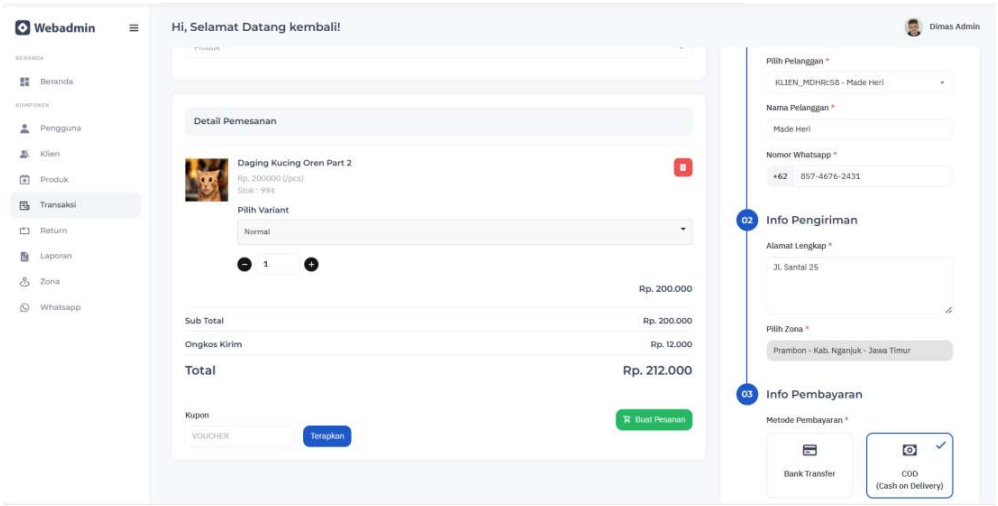


Gambar 4. 24 Prototype & Hasil Pengujian



Gambar 4. 25 Prototype & Hasil Pengujian

Metode pembayaran COD dapat digunakan kembali.

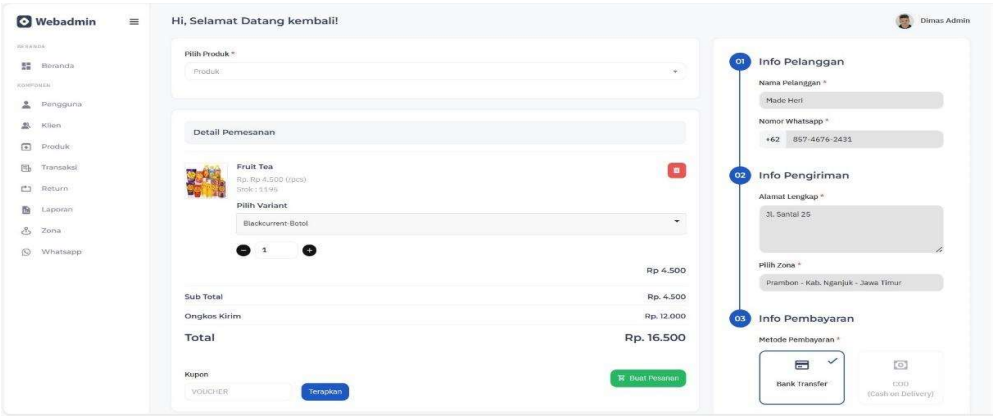


Gambar 4. 26 Prototype & Hasil Pengujian

Tabel 4. 3 Tabel Pengujian Prototype Tahap Ketiga

Skenario	Pengujian	Hasil diharapkan	Hasil pengujian
Melihat dafttar klien	Membuka halaman daftar klien	Berhasil menampilkan daftar klien	**Berhasil**
Setting metode pembayaran	Mengaktifkan kembali fitur pembayaran COD yang dinonaktifkan	Fitur pembayaran COD dapat digunakan lagi dalam proses transaksi	**Berhasil**

Pada tahapan keempat pengembangan aplikasi yaitu berfokus pada fitur kirim ulang pesanan. Admin dapat mengirimkan ulang pesanan pada data transaksi yang sudah diretur. Data barang dari transaksi lama ditampilkan dan dapat diubah.



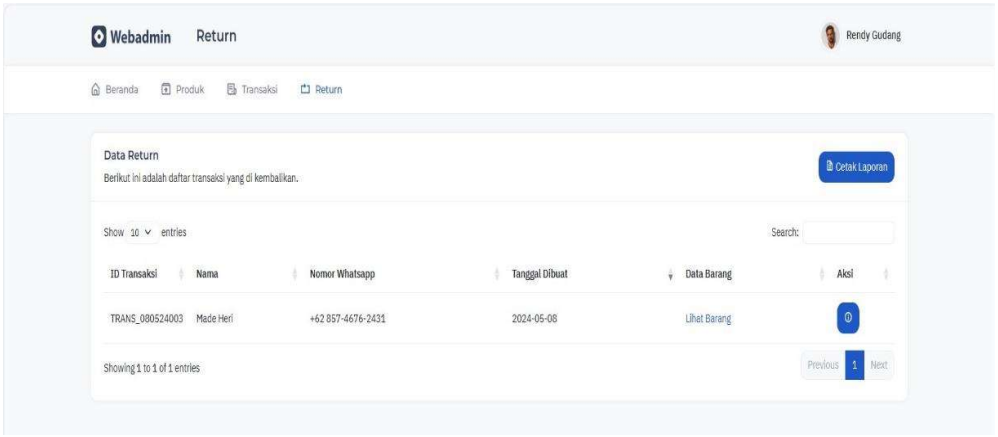
Gambar 4. 27 Prototype & Hasil Pengujian

Tabel 4. 4 Tabel Pengujian Prototype Tahap Keempat

Skenario	Pengujian	Hasil diharapkan	Hasil pengujian
Kirim ulang pesanan	mengirimkan ulang pesanan yang sudah di retur	Data barang dari transaksi lama ditampilkan dan dapat diubah. Berhasil mengirimkan ulang pesanan.	**Berhasil**

b. Gudang

Pada tahapan kelima pengembangan aplikasi yaitu berfokus pada fitur pengelolaan data barang retur oleh role gudang. Gudang berhak mengakses halaman retur untuk melakukan cek kondisi barang sesuai dengan kondisi fisik setelah dilakukan pengecekan fisik barang.



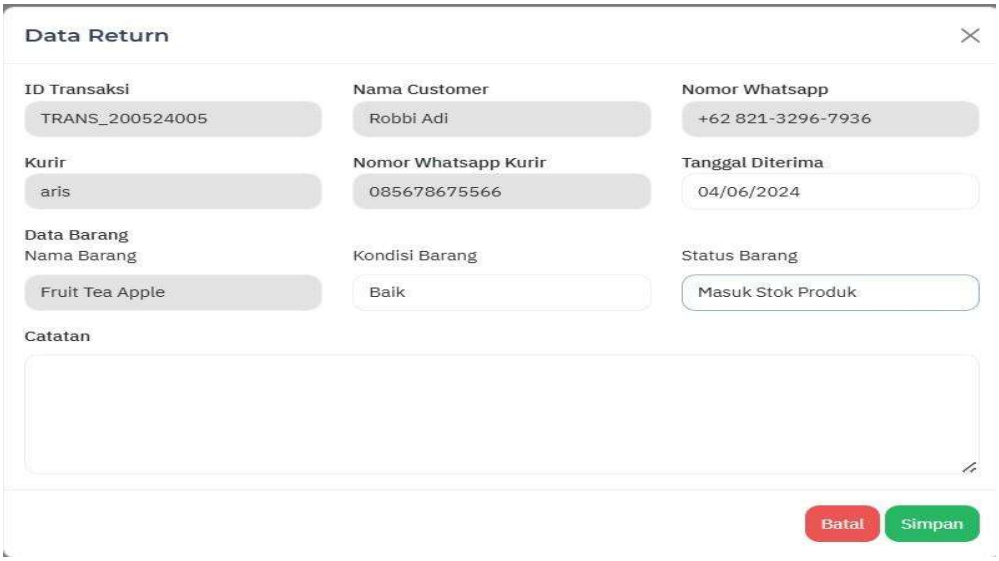
Gambar 4. 28 Prototype & Hasil Pengujian

Berikut adalah hasil pengujian untuk fitur cek kondisi barang :

Kondisi stok produk dan stok retur sebelum dilakukan pengecekan.

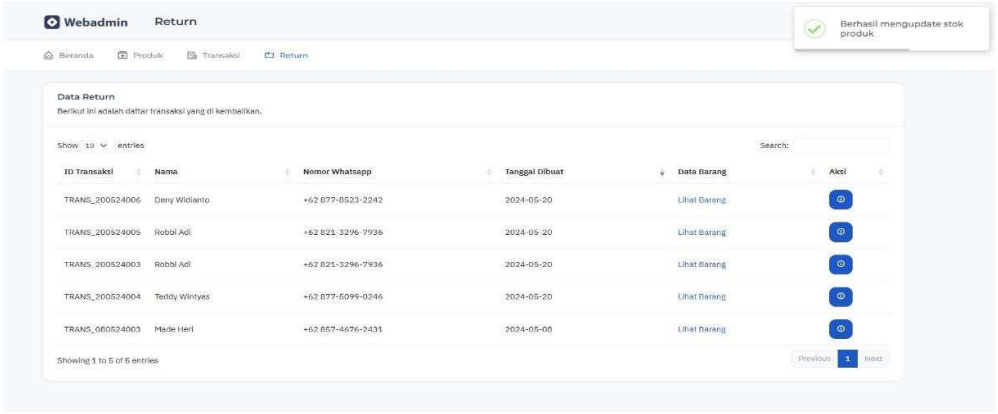


Gambar 4. 29 Prototype & Hasil Pengujian



Gambar 4. 30 Prototype & Hasil Pengujian

Menampilkan message berhasil mengupdate stok produk.



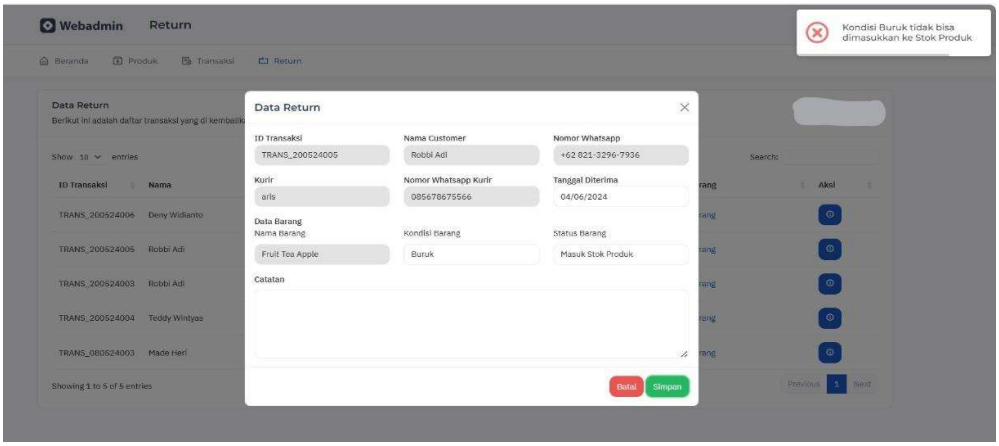
Gambar 4. 31 Prototype & Hasil Pengujian

Kondisi stok produk dan stok retur setelah dilakukan pengecekan.



Gambar 4. 32 Prototype & Hasil Pengujian

Tahapan selanjutnya adalah pengecekan fitur edit kondisi barang. Berdasarkan evaluasi yang dilakukan oleh user, sistem akan tetap mengupdate stok meskipun kondisi barang dan status barang tidak sesuai. Setelah perbaikan, sistem akan menampilkan message gagal jika kondisi barang dan status barang tidak sesuai dan tidak akan mengupdate stok.



Gambar 4. 33 Prototype & Hasil Pengujian

Jika berhasil akan mengupdate stok produk dan stok retur di database. Berdasarkan evaluasi user sebelum perbaikan, merubah data kondisi barang akan menambahkan jumlah stok terus menerus. Setelah perbaikan stok hanya akan update sesuai data yang diubah.



Gambar 4. 34 Prototype & Hasil Pengujian

Tabel 4. 5 Tabel Pengujian Prototype Tahap Kelima

Skenario	Pengujian	Hasil diharapkan	Hasil pengujian
Melihat daftar retur	Membuka halaman daftar retur	Berhasil menampilkan daftar retur	**Berhasil**
Cek kondisi barang	Mengisi data kondisi barang sesuai dengan kondisi fisik	Jika kondisi baik masuk stok produk, jika kondisi buruk masuk stok retur	**Berhasil**
Cek kondisi barang	Mengisi data kondisi barang sesuai dengan kondisi fisik	Jika kondisi baik masuk stok stok retur, jika kondisi buruk masuk stok produk akan menampilkan pesan gagal	**Berhasil**
Edit data retur	Mengubah data kondisi barang yang sudah dicek	Mengupdate data stok barang sesuai kondisi	**Berhasil**

Pada tahapan keenam pengembangan aplikasi yaitu berfokus pada fitur cetak laporan retur. Role gudang dapat melakukan cetak laporan retur per periode.



Gambar 4. 35 Prototype & Hasil Pengujian

	A	B	C	D	E	F	G
1	ID Transaksi	Nama Klien	Nama Kuri	Tanggal Order	Pesanan	Tanggal Diterima	Catatan
2	TRANS_200524006	Deny Widianto	-	2024-05-20	[{"Nama Produk":"Daging Kucing Oren Part 2","Qty":1,"Kondisi":"-","Status":"-"}]	-	-
3	TRANS_200524003	Robbi Adi	aris	2024-05-20	[{"Nama Produk":"Fruit Tea Apple","Qty":1,"Kondisi":"Baik","Status":"Masuk Stok Produk"}]	2024-05-20	-
4	TRANS_200524005	Robbi Adi	aris	2024-05-20	[{"Nama Produk":"Fruit Tea Apple","Qty":1,"Kondisi":"Buruk","Status":"Masuk Stok Retur"}]	2024-06-04	-
5	TRANS_200524004	Teddy Wintyas	aris	2024-05-20	[{"Nama Produk":"Daging Kucing Oren Part 2","Qty":1,"Kondisi":"-","Status":"-"}]	2024-05-20	-
6	TRANS_080524003	Made Heri	aris	2024-05-08	[{"Nama Produk":"Fruit Tea","Qty":1,"Kondisi":"-","Status":"-"}]	2024-05-08	-
7							

Gambar 4. 36 Prototype & Hasil Pengujian

Tabel 4. 6 Tabel Pengujian Prototype Tahap Keenam

Skenario	Pengujian	Hasil diharapkan	Hasil pengujian
Cetak Laporan	Mencetak laporan berdasarkan parameter tanggal	Berhasil mencetak data retur berdasarkan tanggal dalam format excel	**Berhasil**