BAB 4
IMPLEMENTASI SISTEM

Bab ini akan membahas tentang implementasi sistem sesuai dengan hasil analisis dan desain sistem. Implementasi sistem meliputi pemasangan teknologi yang dibutuhkan, pengaturan database, dan implementasi sistem.

4.1. Pemasangan PHP Framework

Sistem ini menggunakan *laravel framework* versi 5.7. Langkah awal proses ini yaitu dengan melakukan pemasangan dan konfigurasi lokal server. Server yang digunakan setidaknya memiliki versi PHP>=7.1.3, ekstensi *PHP*, *JSON*, *OpenSSL*, *Mbstring*, *Tokenizer*, dan *XML*. Untuk melakukan pengecekan dapat memasukan code "<?php phpinfo() ?>" pada file index.php dalam folder public.

4.1.1. Pemasangan Laravel

Tahapan dalam pemasangan *laravel framework* sebagai berikut :

1. Pasang komponen yang dibutuhkan
   - Pastikan composer telah terpasang, lalu inputkan
   - `Composer global require laravel/installer`
   - `Laravel new gudangpmi`
   - pada *command line* windows
2. Download composer
   - Buka https://getcomposer.org/download/ pada browser
   - Download Composer-Setup.exe
   - Terdapat juga command-line installation pada https://getcomposer.org/download/
3. *Running*

```
Owen@Owen-PC MINGW64 /c/xampp/htdocs/gudangpmi
$ php artisan serve
Laravel development server started: <http://127.0.0.1:8000>
```

Gambar 4.1. *Local Development Server*

Karena laravel telah terpasang global pada lokal komputer, untuk menjalankannya dapat menggunakan perintah "*php artisan serve*" pada command line.

4.2. Pengaturan Database

Database diatur menggunakan XAMPP, yaitu sebuah aplikasi server lokal *cross platform* dengan phpmyadmin yang termasuk di dalamnya sebagai *User Interface* untuk akses basis data *MySQL*. Laravel juga menyediakan pengembangan lokal server sehingga memudahkan pengaturan *url* pada browser, lihat gambar 4.1.

4.2.1. Koneksi Database

Laravel menyediakan *file .env* (Segmen Program 4.1) untuk mengatur koneksi *database.* Aplikasi ini menggunakan *database* "*mysql*" dengan nama *database* "gudangpmi", *username* "*root*" dan *password*. *Session_lifetime* memiliki nilai *default* 120 dalam menit. Artinya jika tidak ada aktivitas dalam 120 menit session akan dihapus secara otomatis. Nilai *session_lifetime* dapat diubah sesuai kebutuhan.

Segmen Program 4.1. Koneksi *Database*

```
APP_NAME=Laravel
APP_ENV=local
APP_KEY=[redacted]
APP_DEBUG=true
APP_URL=http://localhost

LOG_CHANNEL=stack

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=gudangpmi
DB_USERNAME=root
DB_PASSWORD=[redacted]

BROADCAST_DRIVER=log
CACHE_DRIVER=file
QUEUE_CONNECTION=sync
SESSION_DRIVER=file
SESSION_LIFETIME=120
```

4.2.2. Membuat Tabel pada Database

Laravel menyediakan berbagai fitur menarik. Pada proses pengembangan aplikasi ini fitur yang digunakan diantaranya *artisan*, *migration*, dan *eloquent*. *Artisan* merupakan sekumpulan perintah yang dapat dijalankan pada *command-line* untuk membuat *model*, *controller*, *migration*, dan sebagainya. Untuk membuat tabel baru pada database menggunakan perintah `php artisan make:migration create_items_table`. Secara otomatis file create_items_table akan dibuat dan berisi fungsi yang dapat dijalankan untuk menambahkan tabel *item* pada *database*. Pada Segmen Program 4.2. Membuat Tabel *Item*, terdapat fungsi yang memuat atribut yang akan diinputkan pada database. Kemudian ketika perintah `php artisan migrate` dijalankan secara otomatis tabel akan dibuat pada *database* dan dapat diakses melaui *phpmyadmin*.

Segmen Program 4.2. Membuat Tabel *Item*

```
<?php
use Illuminate\Support\Facades\Schema;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Database\Migrations\Migration;
class CreateItemsTable extends Migration
{
    public function up()
    {
        Schema::create('items', function (Blueprint $table) {
            $table->increments('id');
            $table->string('ctn', 100);
            $table->string('deskripsi_barang');
            $table->integer('nomor_katalog');
            $table->string('um',50);
            $table->string('ump',50);
            $table->smallInteger('unit_per_ump');
            $table->smallInteger('volume_unit');
            $table->smallInteger('bobot_unit');
            $table->timestamps();
        });
    }
    public function down()
    {
        Schema::dropIfExists('items');
    }
}
```

4.3. Implementasi Program

Aplikasi ini dibangun berdasarkan konsep MVC(*Model, View, Controller*) dengan *laravel php framework*. Implementasi program terdiri dari beberapa tahap, yaitu pembuatan *route*, *controller*, *model*, *view*, dan *class helper*.

4.3.1. Route

Pada laravel pembuatan *route* (Segmen Program 4.3) bertujuan untuk mengarahkan url (*uniform resource locator*) ke halaman yang diinginkan. Dengan kata lain, *route* merupakan daftar halaman.

Segmen Program 4.3. *Route*

```
<?php

Route::get('/', 'AuthController@login')->name('login');
Route::get('login', 'AuthController@login')->name('login');
Route::post('checklogin', 'AuthController@checklogin');
Route::get('logout', 'AuthController@logout');

Route::group(['middleware' => ['auth','checkRole:superuser']], function (){
    Route::get('/user', 'UserController@index');
    Route::post('/user/create', 'UserController@create');
    Route::delete('/user/{id}', 'UserController@destroy');
    /*
        Master Data
    */
    //#Donors
    Route::get('/masterdata/donor', 'MasterController@donor');
    Route::post('/masterdata/donor/create', 'MasterController@createDonor');
    Route::delete('/masterdata/donor/{id}',
'MasterController@destroyDonor');
    //#Warehouse
    Route::get('/masterdata/warehouse', 'MasterController@warehouse');
    Route::post('/masterdata/warehouse/create',
'MasterController@createWH');
    Route::delete('/masterdata/warehouse/{id}',
'MasterController@destroyWH');
    //#Items
    Route::delete('/masterdata/list/{id}', 'ListItemController@destroy');
    //GroupItem code
    Route::get('masterdata/groupitem','GroupController@index');
    Route::post('/masterdata/groupitem/create', 'GroupController@create');
    Route::get('/masterdata/groupitem/edit', 'GroupController@edit');
    Route::get('/masterdata/groupitem/{id}/delete',
'GroupController@destroy');
    //FamilyItem code
    Route::get('masterdata/familyitem','FamilyController@index');
    Route::post('/masterdata/familyitem/create', 'FamilyController@create');
    Route::get('/masterdata/familyitem/edit', 'FamilyController@edit');
    Route::get('/masterdata/familyitem/{id}/delete',
'FamilyController@destroy');
    //#Project
    Route::get('/masterdata/project', 'ProjectController@index');
    Route::post('/masterdata/project/create', 'ProjectController@create');
```

```
    Route::delete('/masterdata/project/{id}', 'ProjectController@destroy');
    //#Transport
    Route::get('/masterdata/transport', 'MasterController@transport');
    Route::post('/masterdata/transport/create',
'MasterController@createTransport');
    Route::delete('/masterdata/transport/{id}',
'MasterController@destroyTransport');
    /*
        Stok
    */
    //#Stock
    Route::get('/masterdata/item/{id}/edit', 'ItemController@edit');
    Route::post('/masterdata/item/{id}/update', 'ItemController@update');
    Route::delete('/masterdata/item/{id}', 'ItemController@destroy');
    Route::post('/masterdata/item/import', 'ItemController@importexcel')-
>name('masterdata.item.import');
    /*
        Berita
    */
    //#News
    Route::get('/masterdata/news/edit', "NewsController@edit");
    Route::post('/masterdata/news/{id}/update', 'NewsController@update');
    Route::get('/masterdata/news/{id}/destroy', 'NewsController@destroy');
    Route::delete('/masterdata/news/{id}', 'NewsController@destroy');

    //#Setting
    Route::get('/masterdata/settings', 'MasterController@index');
    /* Laporan */
});

Route::group(['middleware' =>
['auth','checkRole:superuser,operator,volunteer']], function (){
    Route::get('/masterdata', 'ItemController@index');
    Route::get('/masterdata/item', 'ItemController@index');
});

Route::group(['middleware' => ['auth','checkRole:superuser,operator']],
function (){
    //Stok
    Route::post('/masterdata/item/create', 'ItemController@create');
    //List Daftar Kode
    Route::get('/masterdata/list', 'ListItemController@index');
    Route::post('/masterdata/list/create', 'ListItemController@create');
    /*
        News
    */
    Route::get('/masterdata/news/confirmnews',
"NewsController@showConfirmPage");
    Route::post('/masterdata/news/{id}/confirm', 'NewsController@confirm');
    /*
        Dokumen
    */
    //Waybill-in
    Route::get('/document/receive', "ReceiveDocController@index");
    Route::get('/document/receive/list', "ReceiveDocController@show");
    Route::patch('/document/receive/details',
"ReceiveDocController@showDetail");
    Route::post('/document/receive/create', "ReceiveDocController@create");
    Route::get('/document/receive/{id}/edit', 'ReceiveDocController@edit');
    Route::patch('/document/receive/{id}', 'ReceiveDocController@update');
    Route::delete('/document/receive/{id}', 'ReceiveDocController@destroy');
    Route::get('/document/receive/{id}/printpreview',
'ReceiveDocController@print');
```

```
    Route::get('/document/receive/{id}/export',
'ReceiveDocController@exportPdf');

    //Waybill-out
    Route::get('/document/send', "SendDocController@index");
    Route::post('/document/send/create', "SendDocController@create");
    Route::get('/document/send/list', "SendDocController@show");
    Route::get('/document/send/{id}/edit', 'SendDocController@edit');
    Route::patch('/document/send/{id}', 'SendDocController@update');
    Route::delete('/document/send/{id}', 'SendDocController@destroy');
    Route::get('/document/send/{id}/printpreview',
'SendDocController@print');
    Route::get('/document/send/{id}/export', 'SendDocController@exportPdf');

});

Route::group(['middleware' => ['auth','checkRole:superuser,donor']],
function (){
    Route::get('/itemneeds', 'ItemController@showItemNeeds');
});

Route::group(['middleware' => ['auth','checkRole:superuser,volunteer']],
function (){
    //Request dari Stok
    Route::get('/masterdata/item/sendmessage',
"ItemController@sendMessage"); //request barang
    //Send News
    Route::post('/masterdata/news/create', "NewsController@create");
    //Bisa lihat informasi stok gudang dan berita bencana serta insert
berita bencana
    Route::get('/masterdata/item/sendmessage',
"ItemController@sendMessage");
});

Route::group(['middleware' =>
['auth','checkRole:superuser,operator,donor,volunteer']], function (){
    Route::get('/dashboard', 'DashboardController@index');
    //Profil
    Route::patch('/user/profile', 'UserController@profile');
    Route::patch('/user/edit','UserController@edit'); //update foto profil
    Route::post('/user/{id}/update', 'UserController@update');
    Route::patch('/user/settings', 'UserController@setting'); //Update
Password
    //News
    Route::get('/masterdata/news', "NewsController@index");
});
```

Setelah membuat *route* selanjutnya adalah membuat *controller* yang berisi fungsi – fungsi pada program yang berhubungan langsung dengan *view*. Model juga dibuat sehingga *controller* dapat beriteraksi dengan *database*.

4.3.2. Authentication

Pada bagian (Segmen Program 4.4) *authentication* terdapat fungsi *__construct* sehingga hanya pengguna terdaftar yang dapat masuk ke sistem. Fungsi login untuk menampilkan halaman *login.blade* dan *checklogin* untuk melakukan pengecekan email dan password yang diinputkan terhadap data pengguna pada *database*. Metode *auth* dan *attempt* merupakan metode bawaan laravel. Setelah pengecekan dilakukan, jika email dan password cocok maka halaman akan di *redirect* ke */dashboard*. Kemudian fungsi berikutnya adalah *logut* yang memungkinkan pengguna untuk keluar dari sistem.

Segmen Program 4.4. *Authentication*

```
<?php

namespace App\Http\Controllers;
use Auth;
use Validator;
use Illuminate\Http\Request;

class AuthController extends Controller
{
    public function __construct()
    {
        $this->middleware('guest')->except('logout');
    }
    public function login()
    {
        return view('auths.login');
    }
    public function checklogin(Request $request)
    {
        $this->validate($request, [
            'email'         =>  'required|email',
            'password'      =>  'required',
        ]);

        if(Auth::attempt($request->only('email', 'password'))){
            return redirect('/dashboard');
        }else{
          return back()->with('error', 'email atau password tidak cocok!');
        }
    }
    public function logout()
    {
        Auth::logout();
        return redirect('/login');
}}
```

4.3.3. Pengguna

4.3.3.1. Menampilkan Tabel User

Dengan mengisi variabel *fillable* menggunakan nama kolom pada table, artinya hanya nama kolom yang ditulis yang dapat diisi pada database. Cara akses database ini merupakan teknik *eloquent orm* yang dimiliki laravel. Lihat "Segmen Program 4.5. Lihat halaman pengguna pada *User*", seluruh field tabel *users* dapat langsung diakses dengan membuat *model user*. Data *user* akan ditampung dalam variable *$data_user* yang kemudian akan dirim ke *view* untuk ditampilkan. Seperti pada Segmen Program 4.6. Lihat halaman pengguna pada *UserController*.

Segmen Program 4.5. Lihat halaman pengguna pada *Model User*

```
<?php

namespace App;
use Auth;

class User extends Authenticatable
{
    use Notifiable;
    protected $fillable = [
        'first_name','last_name','gender','religion',
        'address','avatar', 'email', 'password', 'role', 'phone'
    ];
```

Segmen Program 4.6. Lihat halaman pengguna pada *UserController*

```
<?php

namespace App\Http\Controllers;
use App\User;

class UserController extends Controller
{
    public function index()
    {
        $data_user = User::all();
        $data_user = $data_user->sortBy('role');
        return view('user.index', ['data_user' => $data_user]);
    }}
```

4.3.3.2. Menambahkan User Baru

Hanya admin yang dapat menambahkan pengguna, lihat Segmen Program 4.3. Route. Admin dapat menambahkan pengguna beserta seluruh datanya dan memberikan *default password* kepada pengguna. *Password* yang diberikan telah dienkripsi menggunakan fungsi *hashing bycript* seperti pada Segmen Program 4.7. Menambahkan pengguna pada *UserController* di bawah ini.

Segmen Program 4.7. Menambahkan pengguna pada *UserController*

```
<?php

namespace App\Http\Controllers;
use App\User;

class UserController extends Controller
{
    public function create(Request $request)
    {
        $this->validate($request,[
            'first_name'  => 'required|min:3',
            'email'       => 'required|email|unique:users',
            'phone'       => 'required|min:8|unique:users',
            'gender'      => 'required',
            'address'     => 'required',
            'role'        => 'required',
        ]);
        //insert user baru
        $user = new User;
        $user->first_name    = $request->first_name;
        $user->last_name     = $request->last_name;
        $user->email         = $request->email;
        $user->phone         = $request->phone;
        $user->gender        = $request->gender;
        $user->address       = $request->address;
        $user->religion      = $request->religion;
        $user->role          = $request->role;
        $user->password      = bcrypt('$request->password');
        $user->remember_token = str_random(60);
        $user->save();
       return redirect('/user')->with('sukses', 'User berhasil ditambahkan
       ..');
    }
```

4.3.3.3. Menyunting User

Admin juga dapat menyunting data pengguna, diantaranya dengan mengupdate data pengguna serta menambahkan foto profil pengguna seperti pada Segmen Program 4.8. Menyunting pengguna pada *UserController*. Selain itu, admin juga dapat menghapus pengguna lihat Segmen Program 4.9. Menghapus pengguna pada *UserController*.

Segmen Program 4.8. Menyunting pengguna pada *UserController*

```
<?php

namespace App\Http\Controllers;
use App\User;

class UserController extends Controller
{
    public function update(Request $request, $id)
    {
        $this->validate($request,[
            'first_name' => 'required|min:3',
            'email' => ['required',
                        'email',
                        Rule::unique('users')->ignore($request->id),
            ],
            'avatar' => 'image|mimes:jpg,png,jpeg',
        ]);
        //Update User
        $user = User::find($id);
        $user->update($request->all());

        if($request->hasFile('avatar')){
            $avatar = $request->file('avatar');
            $filename = time().'.'.$avatar->getClientOriginalExtension();
            $filenameNavbar = $filename.'_nv'.
       Image::make($avatar)->resize(90,90)->save(
       public_path('superuser/assets/img/'.$filename) );

            $user->avatar = $filename;
            $user->save();
        }
        return redirect('/user')->with('sukses', 'Data berhasil disunting');
    }
```

Segmen Program 4.9. Menghapus pengguna pada *UserController*

```
<?php
```

```
namespace App\Http\Controllers;
use App\User;

class UserController extends Controller
{
    public function destroy($id)
    {
        $user = User::find($id);
        $user->delete($user);
       return redirect('/user')->with('sukseshapus','User "'.$user->
       first_name.'"   berhasil dihapus');
    }
```

4.3.4. Master Data

Pada sistem terdapat tiga master data diantaranya master data stok barang, daftar kode barang dan berita.

4.3.4.1. Menampilkan Stok Barang

Menampilkan stok barang dengan cara mengambil semua data pada *database* menggunakan fitur *eloquent* pada laravel. Data yang diambil disimpan pada *object Item(model)* yang kemudian akan dimasukan ke dalam *variable $data_barang* dan dikirim ke *view*. Tampilan *view* seperti pada Segmen Program 4.11. Menampilkan Tabel Stok pada *View*. Tampilan tabel menggunakan *yajra datatable*. Dengan demikian fitur *pagination* dan *searching* ditambahkan secara otomatis. Lihat segmen program 4.12. Menampilkan Tabel Stok dengan Format yajra Datatable. Pada segmen program tersebut fungsi *DataTable()* dipanggil menggunakan *jQuery* berdasarkan id yang ada pada tabel.

Segmen Program 4.10. Menampilkan Tabel Stok pada *ItemController*

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Item;
class ItemController extends Controller
{
    public function index(Request $request)
    {
          if(Auth::user()->role == 'superuser' || Auth::user()->role ==
'operator'){
             $data_barang = Item::all();
             $kode_barang = Itemcode::all();
             $gudang      = Warehouse::where('check', '1')->firstOrFail();
```

```
    $kode_gudang = $gudang->kode_gudang; $nama_gudang = $gudang->nama_gudang;
             $donor       = Donor::all();

        return view('masterdata.item.index', ['data_barang' => $data_barang,
                                              'kode_barang' => $kode_barang,
                                              'kode_gudang' => $kode_gudang,
                                              'nama_gudang' => $nama_gudang,
                                              'donor'       => $donor]);
          }
          else {
              $data_barang = DB::table('items')
                                    ->where('unit_per_ump', '>', 0)
                                    ->get();
              $kode_barang = Itemcode::all();
              $gudang      = Warehouse::all();
              $donor       = Donor::all();
        return view('masterdata.item.index', ['data_barang' => $data_barang,
                                               'kode_barang' => $kode_barang,
                                               'gudang'      => $gudang,
                                               'donor'       => $donor]);
          }
     }
```

Segmen Program 4.11. Menampilkan Tabel Stok pada *View*

```
<table class="table table-hover" id="stocktable">
   <thead class="thead-dark">
      <tr>
         <th scope="col">CTC</th>
         <th scope="col">Deskripsi Barang</th>
         <th scope="col">UM</th>
         <th scope="col">UMP</th>
         <th scope="col">Unit per UMP</th>
         <th scope="col">Volume Unit</th>
         <th scope="col">Bobot Unit</th>
         @if(Auth()->user()->role == 'superuser')
         <th scope="col">Action</th>
         @endif
      </tr>
   </thead>
   <tbody>
      @foreach($data_barang as $item)
         <tr>
            <td class="text-uppercase">{{$item->ctn}}</td>
            <td>{{$item->deskripsi_barang}}</td>
            <td>{{$item->um}}</td>
            <td>{{$item->ump}}</td>
            <td>{{$item->unit_per_ump}}</td>
            <td>{{$item->volume_unit}}</td>
            <td>{{$item->bobot_unit}}</td>
            <td>
               <div class="form-inline">
                  <div class="form-group">
                     <a href="/masterdata/item/{{$item->id}}/edit" class="btn
                     btn-warning btn-sm"><i class="fa fa-edit" ></i></a>
```

```
                </div>
                <div class="form-group">
                   <form action="/masterdata/item/{{ $item->id }}"
                   method="post">
                   @method('delete')
                   @csrf
                     <button type="submit" class="btn btn-danger btn-sm"
                     onclick="return confirm('Yakin menghapus data?')"><i
                     class="fa fa-eraser" ></i></button>
                   </form>
                </div>
             </div>
          </td>
       </tr>@endforeach</tbody></table>
```

Segmen Program 4.12. Menampilkan Tabel Stok dengan format yajra Datatable

```
<head><link rel="stylesheet" type="text/css"
href="{{asset('DataTables/datatables.min.css')}}"/></head>
...
<script>
       $(document).ready(function(){
              $('#stocktable').DataTable();
       });
</script>
<script src="{{asset('DataTables/datatables.min.js')}}"></script>
```

4.3.4.2. Menambahkan Stok Barang

Menambahkan stok barang dapat dilakukan oleh admin dan operator gudang. Pengguna dapat menambahkan stok dengan menginput data satu per satu (Segmen Program 4.13) atau dengan cara melakukan *import excel* seperti pada Segmen Program 4.14. – 4.16.

Segmen Program 4.13. Menambahkan Stok pada *ItemController*

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\Item;
class ItemController extends Controller
{
    public function create(Request $request)
    {
        $item = new Item;
        $deskripsi = Itemcode::where('kode_barang', $request->kode_barang)-
```

```
>firstOrFail();
        $deskripsi = $deskripsi->deskripsi_barang;
        if (!$request->filled('ctn')) {
            $this->validate($request,[
                'um' => 'required',
                'ump' => 'required',
                'unit_per_ump' => 'required',
            ]);
            $kode_gudang = $request->kode_gudang;
            $date        = date('dmyhis');
            $kode_barang = $request->kode_barang;
            $kode_donor  = $request->kode_donor;
            $ctngen      = $kode_gudang.$date.$kode_barang.$kode_donor;
            //set ctn generator
            $item->ctn   = $ctngen;
        }else{
            $this->validate($request,[
                'ctn' => 'unique:items',
                'um' => 'required',
                'ump' => 'required',
                'unit_per_ump' => 'required',
            ]);
            $item->ctn   = $request->ctn;
        }
        $keterangan = $request->keterangan == NULL ? '' : ' ('.$request-
>keterangan.')';
        $item->deskripsi_barang = $deskripsi.$keterangan;
        $item->um               = $request->um;
        $item->ump              = $request->ump;
        $item->unit_per_ump     = $request->unit_per_ump;
        $item->volume_unit      = $request->volume_unit;
        $item->bobot_unit       = $request->bobot_unit;
        $item->save();
        return redirect('/masterdata/item')->with('sukses', 'Data berhasil
diinput!');
    }
```

Segmen Program 4.14. Menambahkan Stok dengan Import Excel

```
public function importexcel(Request $request)
    {
        $errormsg = 'Tidak berhasil import, cek kembali data';
        $successmsg = 'Data berhasil diinput';
        $item = Item::all();
        $totalitem = $item->count();

        $validator = Validator::make($request->all(), [
            'data_item' => 'required|max:5000|mimes:xlsx,xls,csv',
        ]);

        if($validator->passes()){
            Excel::import(new ItemImport, $request->file('data_item'));
            $newtotal = $item->count();

            if($newtotal == $totalitem){
 return redirect('/masterdata/item')->with('gagalimport', $errormsg);
```

```
            }else{
 return redirect('/masterdata/item')->with('suksesimport', $successmsg);
            }
            dd($validator->errors());
        }
        else if($validator->fails()){
return redirect()->back()->with(['errors'=>$validator->errors()]);
        }
    }
```

Segmen Program 4.15. Form Open File untuk Import Excel

```
{!!Form::open(['route' => 'masterdata.item.import', 'class' => 'form-
horizontal', 'enctype' => 'multipart/form-data'])!!}
{!!Form::file('data_item')!!}
{!! Form::close() !!}
```

Segmen Program 4.16. *Collection* pada Directory Import

```
<?php

namespace App\Imports;

use Illuminate\Support\Collection;
use Maatwebsite\Excel\Concerns\ToCollection;

use App\Item;

class ItemImport implements ToCollection
{
    public function collection(Collection $collection)
    {
        foreach ($collection as $row => $col) {
            if($row > 0){
                if($col[1] != NULL || $col[2] != NULL || $col[3] != NULL ||
                    $col[4] != NULL || $col[5] != NULL){
                    Item::create([
                        'ctn'               => $col[1],
                        'deskripsi_barang'  => $col[2],
                        'um'                => $col[3],
                        'ump'               => $col[4],
                        'unit_per_ump'      => $col[5],
                        //'tanggal' => gmdate('Y-m-d', $tanggal)
                    ]);
                }else{
                    continue; }}
}}}
```

4.3.4.3. Menyunting Stok Barang

Menyunting stok barang dapat dilakukan oleh admin dan operator gudang. Pengguna dapat menyunting stok pertama – tama dengan cara masuk ke halaman edit lalu kemudian meng-*update* data seperti pada segmen perogram 4.17.

Segmen Program 4.17. Menyunting Stok pada *ItemController*

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Item;

class ItemController extends Controller
{
    public function edit($id)
    {
        $item = \App\Item::find($id);
        return view('masterdata.item.edit', ['item' => $item]);
    }
    public function update(Request $request, $id)
    {
        $this->validate($request,[
            'ctn' => 'required',
            'deskripsi_barang' => 'required',
            'um' => 'required',
            'ump' => 'required',
            'unit_per_ump' => 'required',
        ]);
        $item = \App\Item::find($id);
        $item->update($request->all());
        return redirect('/masterdata/item')->with('sukses', 'Data berhasil
disunting');
    }
```

4.3.4.4. Menghapus Stok Barang

Menghapus stok barang dapat dilakukan oleh admin dan operator gudang. Pengguna dapat menghapus stok barang satu per satu sesuai dengan id barang yang dikirim ke fungsi *destroy* seperti pada segmen program 4.18. Menghapus Stok pada *ItemController*.

Segmen Program 4.18. Menghapus Stok pada *ItemController*

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Item;
class ItemController extends Controller
{
    public function destroy($id)
    {
        $item = \App\Item::find($id);
        $item->delete($item);
        return redirect('/masterdata/item')->with('sukseshapus', 'Data
"'.$item->ctn.'" berhasil dihapus');
    }
```

4.3.4.5. Menambahkan Kode Barang

Kode barang pada daftar kode barang langsung ditambahkan secara otomatis dengan mengambil nilai *grup*, *family* dan deskripsi barang seperti pada segmen program 4.19 di bawah ini.

Segmen Program 4.19. Menambahkan Kode Barang pada *ListItemController*

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Validator;
use App\Itemcode;
use App\Groupitem;
use App\Familyitem;

class ListItemController extends Controller
{
    public function create(Request $request)
    {
        $this->validate($request,[
            'deskripsi_barang'  => 'required|min:5',
        ]);
        $itemcode = new Itemcode;
        //Buat kode barang
        $grup     = Groupitem::find($request->groupitem_id);
        $family   = Familyitem::find($request->familyitem_id);

       if (!$request->filled('kode_barang')) {
            $text     = str_replace(' ', '', $request->deskripsi_barang);
```

```
            $depan    = substr($text, 0, 2);
            $text; $count=0;
            $strLength = strlen($text);
            for ($i = 0; $i < $strLength; $i++) {
                $char = $text[$i];
                if (is_numeric($char)) {
                    $tengah = $char;
                    $count++;
                break;}
            }
            if ($count == 1) {$belakang = $tengah.substr($text, -1);}
            else {$belakang = substr($text, -2);}
            $kodebrg  = strtoupper($grup->kode_grup.$family-
            >kode_family.$depan.$belakang);
        }
        else {
            $kodebrg  = strtoupper($grup->kode_grup.$family->
            kode_family.$request->kode_barang); }
        $itemcode->groupitem_id     = $request->groupitem_id;
        $itemcode->familyitem_id    = $request->familyitem_id;
        $itemcode->kode_barang      = $kodebrg;
        $itemcode->deskripsi_barang = $request->deskripsi_barang;
        $itemcode->save();
       return redirect('/masterdata/list')->with('sukses', 'Data berhasil
       diinput!');
    }
}
```

4.3.4.6. Menghapus Kode Barang

Pada bagian ini (Segmen Program 4.20) setiap field dapat dihapus dengan mengirimkan id dari tabel kode barang ke fungsi *destroy* pada *ListItemController*.

Segmen Program 4.20. Menghapus Kode Barang pada *ListItemController*

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Validator;
use App\Itemcode;
class ListItemController extends Controller
{
    public function destroy($id)
    {
        $list = Itemcode::find($id);
        $list->delete($list);
```

```
        return redirect('/masterdata/list')->with('sukseshapus', 'Data
"'.$list->kode_barang.'" berhasil dihapus');
    }
}
```

4.3.4.7. Menampilkan Berita

Segmen Program 4.21. Menampilkan Berita pada *NewsController* di bawah ini menampilkan tiga berita terbaru. Berita yang telah terkonfirmasi akan memiliki nilai *confirm_news* = 1. Seluruh pengguna dapat melihat berita yang telah dibuat dan dikonfirmasi.

Segmen Program 4.21. Menampilkan Berita pada *NewsController*

```
<?php

namespace App\Http\Controllers;
use App\News;
class NewsController extends Controller
{
    public function index()
    {
        $data_news = News::where('confirm_news', 1)
                        ->orderBy('created_at', 'desc')
                        ->paginate(3);
        return view('masterdata.news.index', ['data_news' => $data_news]);
    }
```

4.3.4.8. Menambahkan Berita Bencana

Hanya admin dan relawan yang dapat menambahkan berita. Berita yang ditambahkan oleh admin memiliki nilai “1” sedangkan berita yang ditambahkan oleh relawan memiliki nilai “0”. Nilai “1” akan tampil, sedangkan nilai “0” tidak ditampilkan pada halaman berita.

Segmen Program 4.22. Menambahkan/Mengirim Berita Bencana pada *Controller*

```
<?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\News;
use Auth;
```

```
class NewsController extends Controller
{
   public function create(Request $request)
   {
       $this->validate($request,[
           'title' => 'required|min:5',
           'content_news' => 'required',
       ]);
       $news = new News;
       $news->title       = $request->title;
       $news->content_news= $request->content_news;
       $user = Auth::User();
       if($user->role == 'superuser'){
           $news->confirm_news= 1;
           $news->save();
          return redirect('masterdata/news')->with('sukses', 'Berita
          berhasil ditambahkan!');}
        else {
          $news->confirm_news= 0;
          $news->save();
          return redirect('masterdata/news')->with('sukses', 'Berita
          berhasil ditambah, menunggu konfirmasi ..');}
   }
```

4.3.4.9. Konfirmasi Berita Bencana

Berita yang dikirim oleh relawan akan ditampilkan pada halaman konfirmasi berita. Konfirmasi berita bencana dilakukan oleh admin dengan cara mengubah nilai *variable confirm_news* dari “0” ke “1”. Dengan demikian berita terkirim dapat tampil dan dilihat seluruh pengguna aplikasi. Lihat segmen program 4.23. Konfirmasi Berita pada *NewsController*.

Segmen Program 4.23. Konfirmasi Berita pada *NewsController*

```
<?php

namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\News;

class NewsController extends Controller
{
   public function confirm(Request $request, $id)
   {
       News::where('id', $id)->update(['confirm_news' => 1]);
       $news = News::find($id);
       return redirect('/masterdata/news/confirmnews')-
       >with('sukseskonfirm', 'Berita "'.$news->title.'" dikonfirmasi');
   }
```

4.3.4.10. Menghapus Berita Bencana

Berita yang ada dapat dihapus oleh admin seperti pada segmen program 4.24. Menghapus Berita pada *NewsController*. Halaman berita akan mengirimkan id ke *controller* berdasarkan yang dipilih untuk dihapus.

Segmen Program 4.24. Menghapus Berita pada *NewsController*

```
<?php

namespace App\Http\Controllers;
use Illuminate\Http\Request;
use App\News;

class NewsController extends Controller{
   public function destroy($id){
        $news = News::find($id);
        $news->delete($news);
          return redirect('masterdata/news')->with('sukseshapus', 'Berita
          "'.$news->title.'" berhasil dihapus');
   }
```

4.3.5. Dokumen

Terdapat dua(2) dokumen penting pada gudang yaitu barang masuk dan keluar. *Variable jenis_in_out* akan diisi dengan nilai ‘in’ untuk dokumen barang masuk dan nilai ‘out’ untuk dokumen barang keluar. Lihat segmen program 4.25. Menampilkan Dokumen pada *ReceiveDocController*

4.3.5.1. Menampilkan Dokumen

Berikut menampilkan daftar dokumen masuk seperti pada segmen program 4.25. Menampilkan Dokumen pada *ReceiveDocController*. Data yang ditampilkan berupa tabel.

Segmen Program 4.25. Menampilkan Dokumen pada *ReceiveDocController*

```
<?php

namespace App\Http\Controllers;
use Illuminate\Support\Facades\DB;
use Illuminate\Http\Request;
use App\Document;

class ReceiveDocController extends Controller
{
    public function index()
```

```
    {
        //create dokumen number
        $tahun     = date('y');
        if(Document::where('jenis_in_out', 'in')->count() == 0){
            $nomor_dok = 'IN-'.$tahun.'-'.sprintf("%04d", 1);
        }else {
            $lastdocnum     = Document::where('jenis_in_out', 'in')
            ->orderBy('nomor_dokumen', 'desc')->first()->nomor_dokumen;
            $lastdocnum     = (int)substr($lastdocnum, -4);
            $lastdocnum= sprintf("%04d", $lastdocnum+1);
            $nomor_dok = 'IN-'.$tahun.'-'.$lastdocnum;
        }
        $user      = User::where('role', 'superuser')->orWhere('role',
       'operator')->get();
        $negara         = Countrycode::where('check', '1')->get();
        foreach ($negara as $n) {
            $namanegara = $n->nama_negara;
        }
        $lokasi_terima  = Warehouse::where('check', '1')->get();
        foreach ($lokasi_terima as $lok) {
            $namagudang = $lok->nama_gudang;
        }
        $gudang         = Warehouse::where('check', '0')->get();
        $proyek         = Project::all();
        $donor          = Donor::all();
        $trans          = Transtype::all();
        $item           = Item::all();
        //dd($totaldok,$tahun,$nomor_dok);
        return view('document.receive.index', ['nomor_dok'    => $nomor_dok,
                                        'user'         => $user,
                                        'negara'       => $namanegara,
                                        'lokasi_terima'=> $namagudang,
                                        'gudang'       => $gudang,
                                        'proyek'       => $proyek,
                                        'donor'        => $donor,
                                        'trans'        => $trans,
                                        'item'         => $item]);
    }
```

4.3.5.2. Menampilkan Detail Dokumen

Bagian ini berfungsi menampilkan halaman detail dokumen yang memuat informasi dokumen keluar atau masuk secara detail.

Segmen Program 4.26. *showDetail()* pada *ReceiveDocController*

```
<?php

namespace App\Http\Controllers;
use Illuminate\Support\Facades\DB;
```

```
use Illuminate\Http\Request;
use App\Document;

class ReceiveDocController extends Controller
{
    public function showDetail($id)
    {
        $dokumen    = Document::where('jenis_in_out', 'in')
                              ->find($request->docin_id);
         foreach($dokumen->item as $i){
             $unit   = $i->unit_per_ump;
             $pak    = $i->pivot->pak;
             $non_pak= $i->pivot->unit_tidak_terkemas;
             $jumlah[] = $unit * $pak + $non_pak;
             $jumUnit[]= $unit++;
             $jumPak[] = $pak++;
             $jumNon[] = $non_pak++;
         }
         $totUnit  = array_sum($jumUnit);
         $totalPak = array_sum($jumPak);
         $totalNon = array_sum($jumNon);
         $total    = array_sum($jumlah);
         return view('document.receive.details', ['dokumen' => $dokumen,
                                                  'totunit' => $totUnit,
                                                  'totalpak'=> $totalPak,
                                                  'totalnon'=> $totalNon,
                                                  'total'   => $total]);}
```

4.3.5.3. Memberikan Nomor Dokumen

Pada bagian ini (Segmen Program 4.27) berfungsi memberikan nomor dokumen secara otomatis. Nomor dokumen terdiri dari jenis dokumen, dua digit terakhir tahun dan frekuensi.

Segmen Program 4.27. Memberikan No. Dokumen pada *ReceiveDocController*

```
<?php

namespace App\Http\Controllers;

use Illuminate\Support\Facades\DB;
use Illuminate\Http\Request;

use App\Countrycode;
use App\Warehouse;
use App\Transtype;
use App\Document;
use App\Project;
use App\Donor;
use App\Item;
```

```
class ReceiveDocController extends Controller
{
    public function index()
    {
        //create dokumen number
        $tahun     = date('y');
        if(Document::where('jenis_in_out', 'in')->count() == 0){
            $nomor_dok = 'IN-'.$tahun.'-'.sprintf("%04d", 1);
        }else {
            $lastdocnum     = Document::where('jenis_in_out', 'in')
                              ->orderBy('nomor_dokumen', 'desc')
                              ->first()->nomor_dokumen;
            $lastdocnum     = (int)substr($lastdocnum, -4);
            $lastdocnum= sprintf("%04d", $lastdocnum+1);
            $nomor_dok = 'IN-'.$tahun.'-'.$lastdocnum;
        }
       return view('document.receive.index', ['nomor_dok'
                                          => $nomor_dok]);
    }
```

4.3.5.4. Menambahkan Dokumen

Pada bagian ini (Segmen Program 4.28) dokumen akan ditambahkan dengan fungsi *create* pada *controller*. Pada segmen program sebelum data ditambahkan pada database, pertama – tama dilakukan validasi data, kemudian dilanjutkan dengan menyimpan data baru tabel.

Segmen Program 4.28. Menambahkan Dokumen pada *ReceiveDocController*

```
<?php

namespace App\Http\Controllers;
use Illuminate\Support\Facades\DB;
use Illuminate\Http\Request;
use App\Document;

class ReceiveDocController extends Controller
{
    public function create(Request $request)
    {
        $this->validate($request,[
            'origin'            => 'required',
            'company'           => 'required',
            'alat_angkut'       => 'required',
            'courier_name'      => 'required',
            'trans_condition'   => 'required',
            'date_arrival'      => 'required',
            'user_position' => 'required',
            'user_name'     => 'required',
```

```
            'condition' => 'required'
        ]);
        $negara         = Countrycode::where('check', '1')->get();
        foreach ($negara as $n) {
            $country_id = $n->id;
        }
        $lokasi_terima  = Warehouse::where('check', '1')->get();
        foreach ($lokasi_terima as $lok) {
            $lok_id = $lok->id;
            $lokasi = $lok->nama_gudang;
        }
        $doc = new Document;
        $doc->jenis_in_out   = 'in';
        $doc->countrycode_id = $country_id;
        $doc->warehouse_id   = $lok_id;
        $doc->project_id     = $request->proyek;
        $doc->transtype_id   = $request->alat_angkut;

        $doc->asal           = $request->origin;
        $doc->tujuan         = $lokasi;
        $doc->nomor_dokumen  = $request->nomor_dokumen;
        $doc->komentar       = $request->komentar;
        $doc->jabatan_pengangkut = $request->company;
        $doc->nama_pengangkut= $request->courier_name;
        $doc->kondisi_angkut = $request->trans_condition;
        $date_arrival        = date($request->date_arrival.' h:i:s');
        if($request->date_arrival != date('Y-m-d'))
            $doc->tanggal        = $request->date_arrival;
        else
            $doc->tanggal        = $date_arrival;
        $doc->jabatan        = $request->user_position;
        $doc->nama           = $request->user_name;
        $doc->kondisi        = $request->condition;
        $doc->save();

        $count = count($request->item_id);
        $id  = Document::all()->last()->id;
        //Document::orderBy('id', 'desc')->first()->id;
        $doc = Document::find($id);

        $i = 0;
        while($i <= $count-1) {
            $doc->item()->attach($request->item_id[$i],
                      ['unit' => $request->unit_in_pack[$i],
                       'pak'  => $request->pack[$i],
                       'unit_tidak_terkemas'=>$request->unit_non_pack[$i]]);
            $i++;
        }
        return redirect('/document/receive/list')->with('sukses', 'Dokumen
'.$request->nomor_dokumen.' berhasil ditambahkan!');
    }
```

4.3.5.5. Menyunting Dokumen

Pada bagian sunting dokumen terdapat dua fungsi yaitu pertama untuk menampilkan halaman edit seperti pada Segmen Program 4.29. Menampilkan Sunting Dokumen pada *ReceiveDocController*. Berikutnya yaitu melakukan update data pada *database* seperti pada Segmen Program 4.30. Menyunting Dokumen pada *ReceiveDocController*

Segmen Program 4.29. Edit Dokumen pada *ReceiveDocController*

```
<?php

namespace App\Http\Controllers;
use Illuminate\Support\Facades\DB;
use Illuminate\Http\Request;
use App\Document;

class ReceiveDocController extends Controller
{
    public function edit($id)
    {
        $doc = Document::find($id);
        $negara         = Countrycode::all();
        $lokasi_terima  = Warehouse::all();
        $proyek         = Project::all();
        $donor          = Donor::all();
        $trans          = Transtype::all();
        $item           = Item::all();
        return view('document.receive.edit', ['doc' => $doc,
                                              'negara'       => $negara,
                                    'lokasi_terima'=> $lokasi_terima,
                                              'proyek'       => $proyek,
                                              'donor'        => $donor,
                                              'trans'        => $trans,
                                              'item'         => $item]);
    }
```

Segmen Program 4.30. Menyunting Dokumen pada *ReceiveDocController*

```
<?php

namespace App\Http\Controllers;
use Illuminate\Support\Facades\DB;
use Illuminate\Http\Request;
use App\Document;
```

```
class ReceiveDocController extends Controller
{
    public function update(Request $request, $id)
    {
        $this->validate($request,[
            'origin'            => 'required',
            'company'           => 'required',
            'alat_angkut'       => 'required',
            'courier_name'      => 'required',
            'trans_condition'   => 'required',
            'date_arrival'      => 'required',
            'user_position'     => 'required',
            'user_name'         => 'required',
            'condition'         => 'required'
        ]);
        $doc = Document::find($id);
        if($request->item_id != NULL){
            $countReq = count($request->item_id);
            if($doc->item->count() == 0){
                for($i=0 ; $i <= $countReq-1; $i++) {
                    $doc->item()->attach($request->item_id[$i],
                    ['unit' => $request->unit_in_pack[$i],
                     'pak'  => $request->pack[$i],
                     'unit_tidak_terkemas'=>$request->unit_non_pack[$i]]);
                }
            }else{
                $count = $doc->item->count();
                if($countReq == $count || $countReq > $count){
           for($i=0; $i <= $count-1; $i++) {
              $doc->item()->updateExistingPivot($request->item_id[$i],
                      ['unit' => $request->unit_in_pack[$i],
                       'pak'  => $request->pack[$i],
                       'unit_tidak_terkemas'=>$request->unit_non_pack[$i]]);
                      }
           for($i=$count ; $i <= $countReq-1; $i++) {
              $doc->item()->attach($request->item_id[$i],
                      ['unit' => $request->unit_in_pack[$i],
                      'pak'  => $request->pack[$i],
                      'unit_tidak_terkemas'=>$request->unit_non_pack[$i]]);
                      }
       }else if($countReq < $count){
              return redirect()->back()->with('erroritem', 'Tidak dapat
              mengurangkan jumlah item!');
        }
      }
    }
        $doc->update(['asal'            => $request->origin,
                      'project_id'      => $request->proyek,
                      'transtype_id'    => $request->alat_angkut,
                      'komentar'        => $request->komentar,
                      'nama_pengangkut' => $request->courier_name,
                      'jabatan_pengangkut' => $request->company,
                      'kondisi_angkut'  => $request->trans_condition,
```

```
                        'tanggal'         => $request->date_arrival,
                        'nama'            => $request->user_name,
                        'jabatan'         => $request->user_position,
                        'kondisi'         => $request->condition
                    ]);

        return redirect('/document/receive/list')->with('sukses', 'Dokumen
'.$request->nomor_dokumen.' berhasil disunting');
    }
```

4.3.5.6. Menghapus Dokumen

Pada bagian ini seperti penghapusan yang umum pada sistem dengan mengirimkan id dokumen pada *controller* setelah itu sistem akan menghapus data pada *database*.

Segmen Program 4.31. Menghapus Dokumen pada *ReceiveDocController*

```
<?php

namespace App\Http\Controllers;
use Illuminate\Support\Facades\DB;
use Illuminate\Http\Request;
use App\Document;

class ReceiveDocController extends Controller
{
    public function destroy($id) {
        $doc = Document::find($id);
        $doc->item()->detach([$id]);
        $doc->delete($doc);
        return redirect('/document/receive/list')->with('sukseshapus', 'Data
"'.$doc->nomor_dokumen.'" berhasil dihapus');
    }
```

4.3.5.7. Mencetak Dokumen

Sebelum mencetak dokumen yang akan ditampilkan *preview* untuk cetak dokumen seperti pada Segmen Program 4.32. Menampilkan *Printpreview* pada *ReceiveDocController*. Kemudian pada saat *button* cetak di klik maka fungsi *javascript printme()* akan dipanggil dan melakukan cetak dokumen dengan seperti pada Segmen Program 4.33. *Script* Mencetak dokumen. Dengan menggunakan *window.print()* jendala cetak akan ditampilkan pada browser. Pengguna dapat terlebih dahulu memilih orientasi *potrait* atau *landscape* dan lain sebagainya.

Segmen Program 4.32. Menampilkan *Printpreview* pada *ReceiveDocController*

```
class ReceiveDocController extends Controller
{
    public function print($id)
    {
        $dokumen    = Document::where('jenis_in_out', 'in')->find($id);

        foreach($dokumen->item as $i){
            $unit   = $i->pivot->unit;
            $pak    = $i->pivot->pak;
            $non_pak= $i->pivot->unit_tidak_terkemas;
            $jumlah[] = $unit * $pak + $non_pak;
            $jumUnit[]= $unit++;
            $jumPak[] = $pak++;
            $jumNon[] = $non_pak++;
        }
        $totUnit  = array_sum($jumUnit);
        $totalPak = array_sum($jumPak);
        $totalNon = array_sum($jumNon);
        $total    = array_sum($jumlah);
        return view('document.receive.printpreview', ['dokumen' => $dokumen,
                                                      'totunit' => $totUnit,
                                                      'totalpak'=> $totalPak,
                                                      'totalnon'=> $totalNon,
                                                      'total'   => $total]); }
```

Segmen Program 4.33. *Script* Mencetak dokumen

```
<script>
    function printme(){
        var x = document.getElementById("buttonPrint");
        var close = document.getElementById("buttonClose");
        document.title = '';
        x.style.visibility = "hidden";
        close.style.visibility = "hidden";
        window.print();
        x.style.visibility = "visible";
        close.style.visibility = "visible";
    }
</script>
```

4.3.5.8. Export Dokumen

Pada bagian *export file* digunakan package *barry/laravel-dompdf* dengan cara dipasang pada laravel *framework* menggunakan perintah (`composer require barryvdf/laravel-dompdf`). Kemudian buka file `app.php` pada foler config lalu tambahkan `('PDF' => Barryvdh\DomPDF\Facade::class,)` pada bagian (`'aliases'`) sehingga dapat diakses langsung pada *ReceiveDocController* dengan menggunakan use PDF seperti pada segmen program 4.34. *Export PDF file* pada *ReceiveDocController*, dibawah ini.

Segmen Program 4.34. *Export PDF file* pada *ReceiveDocController*

```
<?php

namespace App\Http\Controllers;

use Illuminate\Support\Facades\DB;
use Illuminate\Http\Request;
use App\Document;
use PDF;

class ReceiveDocController extends Controller {
    public function exportPdf($id){
        $dokumen    = Document::where('jenis_in_out', 'in')->find($id);
        foreach($dokumen->item as $i){
            $unit   = $i->pivot->unit;
            $pak    = $i->pivot->pak;
            $non_pak= $i->pivot->unit_tidak_terkemas;
            $jumlah[] = $unit * $pak + $non_pak;
        }
        $total    = array_sum($jumlah);

        $pdf = PDF::loadView('document.receive.exportpdf',
                             ['dokumen' => $dokumen,
                              'total'   => $total]);
        return $pdf->download('dokumenmasuk.pdf');
    }
```

4.3.6. Laporan

Terdapat beberapa laporan yang akan dibuat diantaranya melihat laporan stok barang per tahun, melakukan export stok barang tahunan, mencetak laporan stok barang, membuat dan mencetak kartu stok, terakhir melihat daftar kebutuhan barang.

4.3.6.1. Menampilkan Laporan

Laporan stok gudang tahunan dan kartu stok dapat ditampilkan seperti pada segmen program 4.35. Menampilkan Laporan Stok pada *ReportController* dan segmen program 4.36. Menampilkan Kartu Stok pada *ReportController*.

Segmen Program 4.35. Menampilkan Laporan Stok pada *ReportController*

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

use App\Item;

class ReportController extends Controller
{
    public function showStockReport(){
        $data_barang = Item::all();
        return view('itemneeds.index', ['data_barang' => $data_barang]);
    }
```

Segmen Program 4.36. Menampilkan Kartu Stok pada *ReportController*

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

use App\Item;

class ReportController extends Controller
{
    public function showStockCard(){
        $data_barang = Item::all();
        return view('itemneeds.index', ['data_barang' => $data_barang]);
    }
```

4.3.6.2. Menampilkan Daftar Kebutuhan Barang

Pada bagian ini menampilkan kebutuhan barang yang dapat dilihat oleh pengguna dengan peran sebagai donatur. Hanya stok barang yang kurang dari 10 yang akan ditampilkan dan dilihat oleh donatur.

Segmen Program 4.37. Menampilkan kebutuhan barang pada *ItemController*

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;

use App\Item;

class ItemController extends Controller
{
    public function showItemNeeds(){
        $data_barang = DB::table('items')
                                ->where('unit_per_ump', '<', 10)
                                ->get();
        return view('itemneeds.index', ['data_barang' => $data_barang]);
    }
```