**APPENDIX**

1. **RELEVANT SOURCE CODE**

<?php

namespace App\Http\Controllers;

use App\Service;

use Illuminate\Http\Request;

use App\TransactionDetail;

use App\TransactionHeader;

use App\Doctor;

use App\User;

use App\Patient;

use Auth;

use DataTables;

use URL;

use Carbon\Carbon;

use Vinkla\Hashids\Facades\Hashids;

class AppointmentController extends Controller

{

public static $master\_name = 'Appointment';

public static $master\_path = '/appointment';

public static $header\_list = ['DOCTOR', 'APPOINTMENT FROM', 'APPOINTMENT TO', 'STATUS', 'ACTION'];

public function showPatientAppointment()

{

$param['action'] = 'index';

$form\_obj = $this->getFormObject($param);

$user\_detail = Auth::user();

$patient\_detail = Patient::where('patient\_id', $user\_detail->id)->first()

if ($patient\_detail == null) {

$param['action'] = 'create';

$form\_obj = $this->getFormObject($param);

// Override form generic objects

$form\_obj['card\_title'] = 'PATIENT PROFILE INFORMATION';

$form\_obj['form\_action'] = '/patient/profile';

$detail\_collection = Auth::user();

$detail\_collection->email\_address = $detail\_collection->email;

return view('patient.user\_patient\_details', compact('form\_obj', 'detail\_collection'))

->with('alert\_message', 'Please complete your personal information first.');

}

$patient\_id = $patient\_detail->id;

return view('patient.appointment', compact('form\_obj', 'user\_detail', 'patient\_id'));

}

public function getFormObject($param)

{

$user\_detail = Auth::user();

$master\_name = self::$master\_name;

$obj['type'] = ($user\_detail->is\_admin == '0') ? 'admin' : 'patient';

$obj['master'] = strtolower(str\_replace(' ', '', self::$master\_name));

$obj['header\_list'] = self::$header\_list;

$obj['back\_to\_list'] = ($user\_detail->is\_admin == '0') ? '/manage/patient' : '/manage/appointment/';

switch ($param['action']) {

case 'index':

$obj['card\_title'] = strtoupper($master\_name) . ' LIST';

$obj['form\_action'] = ($user\_detail->is\_admin == '0') ? '/manage/patient/create' : '/manage/patient/create';

$obj['action'] = 'index';

break;

case 'payment':

$obj['header\_list'] = ['DOCTOR', 'APPOINTMENT FROM', 'APPOINTMENT TO', 'TOTAL AMOUNT', 'STATUS', 'ACTION'];

$obj['card\_title'] = strtoupper($master\_name) . ' LIST';

$obj['form\_action'] = ($user\_detail->is\_admin == '0') ? '/manage/patient/create' : '/manage/patient/create';

$obj['action'] = 'index';

break;

case 'create':

$obj['card\_title'] = 'CREATE NEW ' . strtoupper($master\_name);

$obj['form\_action'] = '/' . $obj['master'];

$obj['action'] = 'create';

break;

case 'edit':

$obj['card\_title'] = 'MANAGE ' . strtoupper($master\_name);

$obj['form\_action'] = '/' . $obj['master'] . '/' . $param['detail\_id'] . '/update';

$obj['action'] = 'edit';

break;

default:

$obj['card\_title'] = '';

$obj['form\_action'] = '';

$obj['action'] = '';

break;

}

return $obj;

}

public function managePatientAppointment()

{

$user\_detail = Auth::user();

if ($user\_detail->is\_admin != 1) {

return redirect('manage/patient');

}

$param['action'] = 'index';

$form\_obj = $this->getFormObject($param);

$user\_detail = Auth::user();

now = Carbon::now();

$month\_start = $now->startOfMonth();

$detail\_collection = TransactionHeader::where('reservation\_from', '>', $month\_start)

->whereIn('status', ['CF', 'P', 'A-P'])

->orderBy('reservation\_from', 'asc')

->get();

foreach ($detail\_collection as $appointment) {

$appointment->day = Carbon::createFromFormat('Y-m-d H:i:s', $appointment->reservation\_from)->day;

$appointment->month = Carbon::createFromFormat('Y-m-d H:i:s', $appointment->reservation\_from)->month;

$appointment->month\_name = Carbon::createFromFormat('Y-m-d H:i:s', $appointment->reservation\_from)->englishMonth;

$appointment->year = Carbon::createFromFormat('Y-m-d H:i:s', $appointment->reservation\_from)->year;

$appointment->patient\_info = Patient::findOrFail($appointment->patient\_id);

$appointment->name = ($appointment->patient\_info->middle\_name == '') ? $appointment->patient\_info->first\_name . ' ' :

$appointment->patient\_info->first\_name . ' ' . $appointment->patient\_info->middle\_name . ' ';

$appointment->name = $appointment->name . $appointment->patient\_info->last\_name;

$reservation\_from = Carbon::parse($appointment->reservation\_from);

$appointment->reservation\_from\_time = $reservation\_from->format('g:i A');

$reservation\_to = Carbon::parse($appointment->reservation\_to);

$appointment->reservation\_to\_time = $reservation\_to->format('g:i A');

}

$detail\_collection = $detail\_collection->groupBy(['year', 'month', 'day']);

foreach ($detail\_collection as $year => $detail) {

foreach ($detail as $month => $appointment\_month) {

foreach ($appointment\_month as $day => $appointment\_days) {

foreach ($appointment\_days as $appointment) {

$appointment\_month->month = $appointment->month\_name;

$appointment->link = URL::to('/appointment/' . $appointment->id . '/edit');

}

}

}

}

return view('patient.patient\_appointment', compact('form\_obj', 'user\_detail', 'detail\_collection'));

}

public function manageAppointment()

{

$user\_detail = Auth::user();

if ($user\_detail->is\_admin != 1) {

return redirect('manage/patient');

}

$param['action'] = 'index';

$form\_obj = $this->getFormObject($param);

$user\_detail = Auth::user();

$patient\_id = 0;

return view('patient.appointment', compact('form\_obj', 'user\_detail', 'patient\_id'));

}

public function manageAppointmentPayment()

{

$user\_detail = Auth::user();

if ($user\_detail->is\_admin != 1) {

return redirect('manage/patient');

}

$param['action'] = 'payment';

$form\_obj = $this->getFormObject($param);

$user\_detail = Auth::user();

$patient\_id = 0;

return view('patient.appointment\_payment', compact('form\_obj', 'user\_detail', 'patient\_id'));

}

public function cancelAppointment($id)

{

$detail\_collection = TransactionHeader::findOrFail($id);

$detail\_collection->status = 'CL';

$detail\_collection->save();

return redirect('manage/appointment')->with('success', 'Appointment is cancelled successfully.');

}

public function confirmAppointment($id)

{

$detail\_collection = TransactionHeader::findOrFail($id);

$detail\_collection->status = 'CF';

$detail\_collection->save();

return redirect('manage/appointment')->with('success', 'Appointment is confirmed successfully.');

}

public function postAppointment($id)

{

$detail\_collection = TransactionHeader::findOrFail($id);

$detail\_collection->status = 'P';

$detail\_collection->save();

return redirect('manage/appointment')->with('success', 'Appointment is posted successfully.');

}

public function voidAppointment($id)

{

$detail\_collection = TransactionHeader::findOrFail($id);

$detail\_collection->status = 'V';

$detail\_collection->save();

return redirect('manage/appointment')->with('success', 'Appointment is voided successfully.');

}

public function updateAppointment(Request $request, $id)

{

$detail\_collection = TransactionHeader::findOrFail($id);

request()->validate([

'reservation\_from' => 'required|date|after:today',

'reservation\_to' => 'required|date|after:reservation\_from',

]);

$detail\_collection->reservation\_from = $request->reservation\_from;

$detail\_collection->reservation\_to = $request->reservation\_to;

if ($request->doctor\_id == 'None') {

$detail\_collection->doctor\_id = null;

} else {

$detail\_collection->doctor\_id = $request->doctor\_id;

}

if ($request->patient\_id == 'None') {

$detail\_collection->patient\_id = null;

} else {

$detail\_collection->patient\_id = $request->patient\_id;

if ($request->patient\_id == null) {

$patient\_detail = Patient::where('patient\_id', Auth::user()->id)->first();

$detail\_collection->patient\_id = $patient\_detail->id;

}

}

$detail\_collection->remarks = $request->remarks;

$detail\_collection->status = 'S';

$detail\_collection->save();

$services = explode(',', $request->service\_id);

$deleted\_rows = TransactionDetail::where('transaction\_header\_id', $detail\_collection->id)->delete();

foreach ($services as $service) {

$service\_detail = new TransactionDetail();

$service\_detail->transaction\_header\_id = $detail\_collection->id;

$service\_detail->service\_id = $service;

$service\_detail->save();

}

return back()->with('success', 'Appointment has been updated successfully.');

}

public function updateAppointmentPayment(Request $request, $id)

{

$detail\_collection = TransactionHeader::findOrFail($id);

$detail\_collection->total\_amount = $request->total\_amount;

$detail\_collection->save();

return back()->with('success', 'Payment has been updated successfully.');

}

public function approveAppointment($id)

{

$detail\_collection = TransactionHeader::findOrFail($id);

$begin = $detail\_collection->reservation\_from;

$end = $detail\_collection->reservation\_to;

$appointment\_collection = TransactionHeader::whereIn('status', array('CF', 'A-P', 'P'))

->where(function ($query) use ($begin, $end) {

$query->where(function ($q) use ($begin, $end) {

$q->where('reservation\_from', '>=', $begin)

->where('reservation\_from', '<', $end);

})->orWhere(function ($q) use ($begin, $end) {

$q->where('reservation\_from', '<=', $begin)

->where('reservation\_to', '>', $end);

})->orWhere(function ($q) use ($begin, $end) {

$q->where('reservation\_to', '>', $begin)

->where('reservation\_to', '<=', $end);

})->orWhere(function ($q) use ($begin, $end) {

$q->where('reservation\_from', '>=', $begin)

->where('reservation\_to', '<=', $end);

});

})->count();

if ($appointment\_collection > 0) {

return redirect('manage/appointment')->with('success', 'Appointment date is already taken. Please check your schedule.');

}

$detail\_collection->status = 'A-P';

$detail\_collection->save();

return redirect('manage/appointment')->with('success', 'Appointment is approved successfully.');

}

public function create()

{

$param['action'] = 'create';

$form\_obj = $this->getFormObject($param);

$doctor\_collection = $this->retrieveDoctorList();

$patient\_collection = $this->retrievePatientList();

$service\_collection = $this->retrieveServiceList();

return view('patient.appointment\_details', compact('form\_obj', 'doctor\_collection', 'service\_collection', 'patient\_collection'));

}

public function retrieveDoctorList()

{

$collection = Doctor::orderBy('first\_name')->get();

foreach ($collection as $detail) {

$detail->name = ($detail->middle\_name == '') ? $detail->first\_name . ' ' : $detail->first\_name;

$detail->name = $detail->name . $detail->last\_name;

}

return $collection;

}

public function retrievePatientList()

{

$collection = Patient::orderBy('first\_name')->get();

foreach ($collection as $detail) {

$detail->name = ($detail->middle\_name == '') ? $detail->first\_name . ' ' : $detail->first\_name . ' ' . $detail->middle\_name . ' ';

$detail->name = $detail->name . $detail->last\_name;

}

return $collection;

}

public function retrieveServiceList()

{

$collection = Service::orderBy('name')->get();

return $collection;

}

public function edit(TransactionHeader $transaction\_header, $id)

{

$user\_detail = Auth::user();

$detail\_collection = TransactionHeader::findOrFail($id);

$transaction\_detail = TransactionHeader::findOrFail($id)->transaction\_detail;

$param['action'] = 'edit';

$param['detail\_id'] = $id;

$form\_obj = $this->getFormObject($param);

if ($detail\_collection->status == 'P') {

$form\_obj['back\_to\_list'] = ($user\_detail->is\_admin == '0') ? '/manage/patient' : '/manage/payment';

$form\_obj['form\_action'] = '/appointment/' . $param['detail\_id'] . '/update\_payment';

}

$doctor\_collection = $this->retrieveDoctorList();

$patient\_collection = $this->retrievePatientList();

$service\_collection = $this->retrieveServiceList();

foreach ($patient\_collection as $patient) {

if ($detail\_collection->patient\_id == $patient->id) {

$patient->is\_set = 'selected';

} else {

$patient->is\_set = '';

}

if ($patient->is\_set == 'selected') {

$patient->info = Patient::findOrFail($patient->id);

}

}

foreach ($doctor\_collection as $doctor) {

if ($detail\_collection->doctor\_id == $doctor->id) {

$doctor->is\_set = 'selected';

} else {

$doctor->is\_set = '';

}

}

foreach ($service\_collection as $service) {

foreach ($transaction\_detail as $detail) {

if ($service->id == $detail->service\_id) {

$service->is\_set = 'checked';

break;

} else {

$service->is\_set = '';

}

}

}

return view('patient.appointment\_details', compact('form\_obj', 'detail\_collection', 'doctor\_collection', 'service\_collection',

'patient\_collection'));

}

public function store(Request $request)

{

request()->validate([

'reservation\_from' => 'required|date|after:today',

'reservation\_to' => 'required|date|after:reservation\_from',

]);

$form\_input = $request->all();

$detail\_collection = new TransactionHeader($form\_input);

if ($detail\_collection->doctor\_id == 'None') {

$detail\_collection->doctor\_id = null;

}

if ($detail\_collection->patient\_id == '') {

$patient\_detail = Patient::where('patient\_id', Auth::user()->id)->first();

$detail\_collection->patient\_id = $patient\_detail->id;

} elseif ($detail\_collection->patient\_id == 'None') {

$detail\_collection->patient\_id = null;

}

$detail\_collection->status = 'S';

$detail\_collection->save();

$services = explode(',', $request->service\_id);

// $deleted\_rows = TransactionDetail::where('transaction\_header\_id', $detail\_collection->id)->delete();

foreach ($services as $service) {

$service\_detail = new TransactionDetail();

$service\_detail->transaction\_header\_id = $detail\_collection->id;

$service\_detail->service\_id = $service;

$service\_detail->save();

}

return redirect('manage/appointment')->with('success', 'Appointment is created successfully.');

}

public function retrieveList($id)

{

$user\_detail = Auth::user();

if ($user\_detail->is\_admin == '1') {

$detail\_collection = TransactionHeader::query()

->orderBy('created\_at', 'desc');

} else {

$detail\_collection = TransactionHeader::query()

->where('patient\_id', $id)

->orderBy('created\_at', 'desc');

}

return DataTables::eloquent($detail\_collection)

->addColumn('doctor\_name', function ($list) {

if ($list->doctor\_id != null) {

$doctor\_detail = Doctor::findOrFail($list->doctor\_id);

$doctor\_detail->name = ($doctor\_detail->middle\_name == '') ? $doctor\_detail->first\_name . ' ' : $doctor\_detail->first\_name;

$doctor\_detail->name = $doctor\_detail->name . $doctor\_detail->last\_name;

$doctor\_name = $doctor\_detail->name;

} else {

$doctor\_name = 'None Specified';

}

return $doctor\_name;

})

->addColumn('status\_column', function ($list) {

switch ($list->status) {

case 'S':

$status\_html = '<span class="badge badge-warning">Pending</span>';

break;

case 'A-P':

$status\_html = '<span class="badge badge-info">Approved</span>';

break;

case 'CF':

$status\_html = '<span class="badge badge-success">Confirmed</span>';

break;

case 'P':

$status\_html = '<span class="badge badge-success">Posted</span>';

break;

case 'V':

$status\_html = '<span class="badge badge-danger">Voided</span>';

break;

case 'CL':

$status\_html = '<span class="badge badge-danger">Cancelled</span>';

break;

default:

$status\_html = '<span class="badge badge-danger">Status not found</span>';

break;

}

return $status\_html;

})

->addColumn('action\_column', function ($list) {

$edit\_link = URL::to(self::$master\_path) . '/' . $list->id . '/edit';

$edit\_html = '<button type="button" class="btn btn-outline-primary btn-sm" onclick="window.location.href=`' . $edit\_link . '`">

<i class="fa fa-folder-open-o"></i> View</button>';

$delete\_html = '<button type="button" class="btn btn-outline-danger btn-sm">

<i class="fa fa-trash-o"></i> Delete</button>';

$action\_html = $edit\_html;

return $action\_html;

})

->rawColumns(['action\_column', 'status\_column'])

->toJson();

}

public function retrieveListPayment($id)

{

$user\_detail = Auth::user();

if ($user\_detail->is\_admin == '1') {

$detail\_collection = TransactionHeader::query()

->where('status', 'P')

->orderBy('created\_at', 'desc');

} else {

$detail\_collection = TransactionHeader::query()

->where('status', 'P')

->where('patient\_id', $id)

->orderBy('created\_at', 'desc');

}

return DataTables::eloquent($detail\_collection)

->addColumn('doctor\_name', function ($list) {

if ($list->doctor\_id != null) {

$doctor\_detail = Doctor::findOrFail($list->doctor\_id);

$doctor\_detail->name = ($doctor\_detail->middle\_name == '') ? $doctor\_detail->first\_name . ' ' : $doctor\_detail->first\_name;

$doctor\_detail->name = $doctor\_detail->name . $doctor\_detail->last\_name;

$doctor\_name = $doctor\_detail->name;

} else {

$doctor\_name = 'None Specified';

}

return $doctor\_name;

})

->addColumn('status\_column', function ($list) {

switch ($list->status) {

case 'S':

$status\_html = '<span class="badge badge-warning">Pending</span>';

break;

case 'A-P':

$status\_html = '<span class="badge badge-info">Approved</span>';

break;

case 'CF':

$status\_html = '<span class="badge badge-success">Confirmed</span>';

break;

case 'P':

$status\_html = '<span class="badge badge-success">Posted</span>';

break;

case 'V':

$status\_html = '<span class="badge badge-danger">Voided</span>';

break;

case 'CL':

$status\_html = '<span class="badge badge-danger">Cancelled</span>';

break;

default:

$status\_html = '<span class="badge badge-danger">Status not found</span>';

break;

}

return $status\_html;

})

->addColumn('action\_column', function ($list) {

$edit\_link = URL::to(self::$master\_path) . '/' . $list->id . '/edit';

$edit\_html = '<button type="button" class="btn btn-outline-primary btn-sm" onclick="window.location.href=`' . $edit\_link . '`">

<i class="fa fa-folder-open-o"></i> View</button>';

$delete\_html = '<button type="button" class="btn btn-outline-danger btn-sm">

<i class="fa fa-trash-o"></i> Delete</button>';

$action\_html = $edit\_html;

return $action\_html;

})

->rawColumns(['action\_column', 'status\_column'])

->toJson();

}

}

<?php

namespace App\Http\Controllers;

use Illuminate\Foundation\Bus\DispatchesJobs;

use Illuminate\Routing\Controller as BaseController;

use Illuminate\Foundation\Validation\ValidatesRequests;

use Illuminate\Foundation\Auth\Access\AuthorizesRequests;

class Controller extends BaseController

{

use AuthorizesRequests, DispatchesJobs, ValidatesRequests;

}

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Patient;

use Auth;

class DashboardController extends Controller

{

public function index()

{

$user\_detail = Auth::user();

$patient\_detail = Patient::where('patient\_id', $user\_detail->id)->first();

if ($user\_detail->is\_admin != 1) {

if ($patient\_detail == null) {

return redirect('/manage/patient/profile');

} else {

return redirect('/manage/patient');

}

} else{

return redirect('/manage/appointment');

}

}

}

<?php

namespace App\Http\Controllers;

use App\Doctor;

use Auth;

use DataTables;

use Illuminate\Http\Request;

use URL;

use Vinkla\Hashids\Facades\Hashids;

class DoctorController extends Controller

{

public static $master\_name = 'Doctor';

public static $master\_path = '/doctor';

public static $header\_list = ['FIRST NAME', 'LAST NAME', 'BIRTH DATE', 'GENDER', 'CONTACT NO.', 'LAST UPDATED DATE',

'ACTION'];

public function index()

{

$param['action'] = 'index';

$form\_obj = $this->getFormObject($param);

return view('master\_management.index', compact('form\_obj'));

}

public function getFormObject($param)

{

$master\_name = self::$master\_name;

$obj['master'] = strtolower(str\_replace(' ', '', self::$master\_name));

$obj['header\_list'] = self::$header\_list;

$obj['back\_to\_list'] = self::$master\_path;

switch ($param['action']) {

case 'index':

$obj['card\_title'] = strtoupper($master\_name) . ' LIST';

$obj['form\_action'] = $obj['master'] . '/create';

$obj['action'] = 'index';

break;

case 'create':

$obj['card\_title'] = 'CREATE NEW ' . strtoupper($master\_name);

$obj['form\_action'] = '/' . $obj['master'];

$obj['action'] = 'create';

break;

case 'edit':

$obj['card\_title'] = 'UPDATE EXISTING ' . strtoupper($master\_name);

$obj['form\_action'] = '/' . $obj['master'] . '/' . $param['detail\_id'] . '/update';

$obj['action'] = 'edit';

break;

default:

$obj['card\_title'] = '';

$obj['form\_action'] = '';

$obj['action'] = '';

break;

}

return $obj;

}

public function create()

{

$param['action'] = 'create';

$form\_obj = $this->getFormObject($param);

return view('master\_management.details.doctor\_details', compact('form\_obj'));

}

public function store(Request $request)

{

request()->validate([

'first\_name' => 'required|min:2|max:50',

'last\_name' => 'required|min:2|max:50',

'birth\_date' => 'required',

'phone\_number' => 'numeric',

'mobile\_number' => 'numeric',

'email\_address' => 'required|email',

]);

$form\_input = $request->all();

$detail\_collection = new Doctor($form\_input);

$detail\_collection->save();

return redirect(self::$master\_path)->with('success', 'Doctor is created successfully.');

}

public function edit(Doctor $doctor)

{

$detail\_collection = $doctor;

$param['action'] = 'edit';

$param['detail\_id'] = Hashids::connection('App\Doctor')->encode($detail\_collection->id);

$form\_obj = $this->getFormObject($param);

return view('master\_management.details.doctor\_details', compact('form\_obj', 'detail\_collection'));

}

public function update(Request $request, Doctor $doctor)

{

request()->validate([

'first\_name' => 'required|min:2|max:50',

'last\_name' => 'required|min:2|max:50',

'birth\_date' => 'required',

'phone\_number' => 'numeric',

'mobile\_number' => 'numeric',

'email\_address' => 'required|email',

]);

$detail\_collection = Doctor::find($request->id);

$detail\_collection->first\_name = $request->first\_name;

$detail\_collection->middle\_name = $request->middle\_name;

$detail\_collection->last\_name = $request->last\_name;

$detail\_collection->birth\_date = $request->birth\_date;

$detail\_collection->gender = $request->gender;

$detail\_collection->address = $request->address;

$detail\_collection->phone\_number = $request->phone\_number;

$detail\_collection->mobile\_number = $request->mobile\_number;

$detail\_collection->email\_address = $request->email\_address;

$detail\_collection->save();

return back()->with('success', 'Doctor has been updated successfully.');

}

public function retrieveList()

{

$detail\_collection = Doctor::query();

return DataTables::eloquent($detail\_collection)

->addColumn('action\_column', function ($list) {

$hash\_id = Hashids::connection('App\Doctor')->encode($list->id);

$edit\_link = URL::to(self::$master\_path) . '/' . $hash\_id . '/edit';

$edit\_html = '<button type="button" class="btn btn-outline-primary btn-sm" onclick="window.location.href=`' . $edit\_link . '`">

<i class="fa fa-edit"></i> Edit</button>';

$delete\_html = '<button type="button" class="btn btn-outline-danger btn-sm">

<i class="fa fa-trash-o"></i> Delete</button>';

$action\_html = $edit\_html . '&nbsp;' . $delete\_html;

return $action\_html;

})

->rawColumns(['action\_column'])

->toJson();

}

}

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

class HomeController extends Controller

{

public function \_\_construct()

{

$this->middleware('auth');

}

public function index()

{

return view('dashboard');

}

}

<?php

namespace App\Http\Controllers;

use App\Patient;

use Auth;

use DataTables;

use Illuminate\Http\Request;

use URL;

use Vinkla\Hashids\Facades\Hashids;

class PatientController extends Controller

{

public static $master\_name = 'Patient';

public static $master\_path = '/patient';

public static $header\_list = ['FIRST NAME', 'LAST NAME', 'BIRTH DATE', 'GENDER', 'CONTACT NO.', 'LAST UPDATED DATE',

'ACTION'];

public function index()

{

$param['action'] = 'index';

$form\_obj = $this->getFormObject($param);

return view('master\_management.index', compact('form\_obj'));

}

public function getFormObject($param)

{

$master\_name = self::$master\_name;

$obj['master'] = strtolower(str\_replace(' ', '', self::$master\_name));

$obj['header\_list'] = self::$header\_list;

$obj['back\_to\_list'] = self::$master\_path;

switch ($param['action']) {

case 'index':

$obj['card\_title'] = strtoupper($master\_name) . ' LIST';

$obj['form\_action'] = $obj['master'] . '/create';

$obj['action'] = 'index';

break;

case 'create':

$obj['card\_title'] = 'CREATE NEW ' . strtoupper($master\_name);

$obj['form\_action'] = '/' . $obj['master'];

$obj['action'] = 'create';

break;

case 'edit':

$obj['card\_title'] = 'UPDATE EXISTING ' . strtoupper($master\_name);

$obj['form\_action'] = '/' . $obj['master'] . '/' . $param['detail\_id'] . '/update';

$obj['action'] = 'edit';

break;

default:

$obj['card\_title'] = '';

$obj['form\_action'] = '';

$obj['action'] = '';

break;

}

return $obj;

}

public function create()

{

$param['action'] = 'create';

$form\_obj = $this->getFormObject($param);

return view('master\_management.details.patient\_details', compact('form\_obj'));

}

public function createUserPatient()

{

$param['action'] = 'create';

$form\_obj = $this->getFormObject($param);

// Override form generic objects

$form\_obj['card\_title'] = 'PATIENT PROFILE INFORMATION';

$form\_obj['form\_action'] = '/patient/profile';

$detail\_collection = Auth::user();

$detail\_collection->email\_address = $detail\_collection->email;

return view('patient.user\_patient\_details', compact('form\_obj', 'detail\_collection'))

->with('alert\_message', 'Please complete your personal information first.');

}

public function store(Request $request)

{

request()->validate([

'first\_name' => 'required|min:2|max:50',

'last\_name' => 'required|min:2|max:50',

'address' => 'required',

'birth\_date' => 'required',

'phone\_number' => 'required|numeric',

'mobile\_number' => 'required|numeric',

'email\_address' => 'required|email',

]);

$form\_input = $request->all();

$detail\_collection = new Patient($form\_input);

$detail\_collection->save();

return redirect(self::$master\_path)->with('success', 'Patient has been created successfully.');

}

public function storeUserPatient(Request $request)

{

request()->validate([

'first\_name' => 'required|min:2|max:50',

'last\_name' => 'required|min:2|max:50',

'address' => 'required',

'birth\_date' => 'required',

'phone\_number' => 'required|numeric',

'mobile\_number' => 'required|numeric',

'email\_address' => 'required|email',

]);

$user\_detail = Auth::user();

$form\_input = $request->all();

$detail\_collection = new Patient($form\_input);

$detail\_collection->patient\_id = $user\_detail->id;

$detail\_collection->save();

return redirect('/manage/patient');

}

public function edit(Patient $patient)

{

$detail\_collection = $patient;

$param['action'] = 'edit';

$param['detail\_id'] = Hashids::connection('App\Patient')->encode($detail\_collection->id);

$form\_obj = $this->getFormObject($param);

return view('master\_management.details.patient\_details', compact('form\_obj', 'detail\_collection'));

}

public function editUserPatient(){

$user\_detail = Auth::user();

$patient\_detail = Patient::where('patient\_id', $user\_detail->id)->first();

if ($patient\_detail == null){

return redirect('/manage/patient/profile')->with('alert', 'Hello');

}

$detail\_collection = $patient\_detail;

$param['action'] = 'edit';

$param['detail\_id'] = Hashids::connection('App\Patient')->encode($detail\_collection->id);

$form\_obj = $this->getFormObject($param);

// Override form generic objects

$form\_obj['card\_title'] = 'PATIENT PROFILE INFORMATION';

$form\_obj['form\_action'] = '/manage/patient/profile/' . $param['detail\_id'] . '/update';

return view('patient.user\_patient\_details', compact('form\_obj', 'detail\_collection'));

}

public function update(Request $request, Patient $patient)

{

request()->validate([

'first\_name' => 'required|min:2|max:50',

'last\_name' => 'required|min:2|max:50',

'address' => 'required',

'birth\_date' => 'required',

'phone\_number' => 'required|numeric',

'mobile\_number' => 'required|numeric',

'email\_address' => 'required|email',

]);

$detail\_collection = Patient::find($request->id);

$detail\_collection->first\_name = $request->first\_name;

$detail\_collection->middle\_name = $request->middle\_name;

$detail\_collection->last\_name = $request->last\_name;

$detail\_collection->birth\_date = $request->birth\_date;

$detail\_collection->gender = $request->gender;

$detail\_collection->address = $request->address;

$detail\_collection->phone\_number = $request->phone\_number;

$detail\_collection->mobile\_number = $request->mobile\_number;

$detail\_collection->email\_address = $request->email\_address;

$detail\_collection->save();

return back()->with('success', 'Patient has been updated successfully.');

}

public function updateUserPatient(Request $request, Patient $patient)

{

request()->validate([

'first\_name' => 'required|min:2|max:50',

'last\_name' => 'required|min:2|max:50',

'address' => 'required',

'birth\_date' => 'required',

'phone\_number' => 'required|numeric',

'mobile\_number' => 'required|numeric',

'email\_address' => 'required|email',

]);

$detail\_collection = Patient::find($request->id);

$detail\_collection->first\_name = $request->first\_name;

$detail\_collection->middle\_name = $request->middle\_name;

$detail\_collection->last\_name = $request->last\_name;

$detail\_collection->birth\_date = $request->birth\_date;

$detail\_collection->gender = $request->gender;

$detail\_collection->address = $request->address;

$detail\_collection->phone\_number = $request->phone\_number;

$detail\_collection->mobile\_number = $request->mobile\_number;

$detail\_collection->email\_address = $request->email\_address;

$detail\_collection->save();

return redirect()->back()->with('success', 'Your information has been updated successfully.');

}

public function retrieveList()

{

$detail\_collection = Patient::query();

return DataTables::eloquent($detail\_collection)

->addColumn('action\_column', function ($list) {

$hash\_id = Hashids::connection('App\Patient')->encode($list->id);

$edit\_link = URL::to(self::$master\_path) . '/' . $hash\_id . '/edit';

$edit\_html = '<button type="button" class="btn btn-outline-primary btn-sm" onclick="window.location.href=`' . $edit\_link . '`">

<i class="fa fa-edit"></i> Edit</button>';

$delete\_html = '<button type="button" class="btn btn-outline-danger btn-sm">

<i class="fa fa-trash-o"></i> Delete</button>';

$action\_html = $edit\_html . '&nbsp;' . $delete\_html;

return $action\_html;

})

->rawColumns(['action\_column'])

->toJson();

}

}

<?php

namespace App\Http\Controllers;

use App\Doctor;

use App\Patient;

use App\Service;

use App\TransactionHeader;

use Illuminate\Http\Request;

use Auth;

use DataTables;

use Crypt;

use URL;

use Barryvdh\DomPDF\Facade as PDF;

use Dompdf\Options;

use DB;

use Carbon\Carbon;

class ReportController extends Controller

{

public static $master\_name = 'Appointment';

public static $master\_path = '/appointment';

public function showTransactionReport()

{

return view('report.transaction\_report\_details');

}

public function showPatientHistory()

{

$patient\_collection = $this->retrievePatientList();

return view('report.patient\_history\_details', compact('patient\_collection'));

}

public function retrievePatientList()

{

}

}