LAMPIRAN

1. web.php

```
Route::get('/', function () {
   return view('welcome');
});

Auth::routes();

Route::get('/', 'HomeController@index')->name('home');
//Report
Route::get('/report', 'HomeController@report')->name('report');
// attendance home
Route::post("attendance/home", "AttendanceController@home")-
>name('attendance-home');


// check for logged in user
Route::group(['middleware' => ['auth']], function () {

        Route::get('/dashboard', 'HomeController@dashboard')-
>name('dashboard');

        Route::group(["prefix"=>"user"], function(){

                // user create
                Route::get("create", "UserController@create")-
>name('user-create');

                // ID adalah parameter URL
                // user edit
                Route::get("edit/{id}", "UserController@edit")-
>name('user-edit');

                // user update
                Route::post("update/{id}", "UserController@update")-
>name('user-update');

                // user delete
```

```
                Route::get("delete/{id}", "UserController@delete")-
>name('user-delete');

                // user store
                Route::post("store", "UserController@store")-
>name('user-store');

                // user list
                Route::get("index","UserController@index")-
>name('user-index');

                // brand CRUD
                Route::get("brandindex", "BrandController@index")-
>name('brand-index');
                Route::get("brandcreate", "BrandController@create")-
>name('brand-create');
                Route::post("brandsave", "BrandController@save")-
>name('brand-save');
                Route::get("brandedit/{id}",
"BrandController@edit")->name('brand-edit');
                Route::post("brandeditsave/{id}",
"BrandController@editsave")->name('brand-editsave');
                Route::get("branddelete/{id}",
"BrandController@delete")->name('brand-delete');

                // departemen CRUD
                Route::get("departementindex",
"DepartemenController@index")->name('departement-index');
                Route::get("departementcreate",
"DepartemenController@create")->name('departement-create');
                Route::post("departementsave",
"DepartemenController@save")->name('departement-save');
                Route::get("departementedit/{id}",
"DepartemenController@edit")->name('departement-edit');
                Route::post("departementeditsave/{id}",
"DepartemenController@editsave")->name('departement-editsave');
                Route::get("departementdelete/{id}",
"DepartemenController@delete")->name('departement-delete');

                //company CRUD
```

```
                    Route::get("companyindex",
"CompanyController@index")->name('company-index');
                    Route::get("companycreate",
"CompanyController@create")->name('company-create');
                    Route::post("companysave",
"CompanyController@save")->name('company-save');
                    Route::get("companyedit/{id}",
"CompanyController@edit")->name('company-edit');
                    Route::post("companyeditsave/{id}",
"CompanyController@editsave")->name('company-editsave');
                    Route::get("companydelete/{id}",
"CompanyController@delete")->name('company-delete');


                    //Overtime CRUD
                    Route::get("posisiindex",
"PositionController@index")->name('posisi-index');
                    Route::get("posisicreate",
"PositionController@create")->name('posisi-create');
                    Route::post("posisisave", "PositionController@save")-
>name('posisi-save');
                    Route::get("posisiedit/{id}",
"PositionController@edit")->name('posisi-edit');
                    Route::post("posisieditsave/{id}",
"PositionController@editsave")->name('posisi-editsave');
                    Route::get("posisidelete/{id}",
"PositionController@delete")->name('posisi-delete');
          });


          Route::group(["prefix"=>"attendance"], function(){

                    // generate attendance monthly
                    Route::get("generate/{user_id}",
"AttendanceController@generate")->name('attendance-user-generate');

                    // attendance index

          Route::get('{user_id}',"AttendanceController@index")-
>name('attendance-user-index')->where('user_id', '[0-9]+');
```

```
                // atendance store
                Route::post("store","AttendanceController@store")-
>name('attendance-user-store');

                // attendance edit
                Route::get("edit/{}","AttendanceController@edit")-
>name('attendance-user-edit');

                // attendance update
                Route::post("update/{}",
"AttendanceController@update")->name('attendance-user-update');

                // attendance delete
                Route::get("delete/{}",
"AttendanceController@delete")->name('attendance-user-delete');

                // change schedule
                Route::get("schedule/change/{id?}/{schedule_id?}",
"AttendanceController@changeSchedule")->name('attendance-change-
schedule');

                // breaktime attendance
                Route::get("{id}/breaktime",
"AttendanceController@indexBreaktime")->name('attendance-
breaktime-index');

                // count salary from attendance
                Route::get("{user_id}/salary",
"AttendanceController@countSalary")->name('attendance-salary');

        });
```

2. **Attendancecontroller .php**

```
<?php

namespace App\Http\Controllers;

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

```
use App\Position;
use App\BrandName;
use App\Department;
use App\Company;
use App\User;
use App\Attendance;
use App\Schedule;
use App\Breaktime;
use App\Salary;
use Carbon\Carbon;

class AttendanceController extends Controller
{
   public function generate($user_id)
   {
      $attendance = Attendance::where('user_id', $user_id)-
>orderBy('date', 'desc')->first();

      // if attendance is null, create attendance for this month
      if(!$attendance)
      {
         $totalDaysInMonth = Carbon::now()->daysInMonth;
         $month = Carbon::now()->month;

         $attendanceArray = [];
         for($i=1;$i<=$totalDaysInMonth;$i++)
         {
            $attribute['schedule_id'] = 1;
            $attribute['user_id'] = $user_id;
            $attribute['date'] = Carbon::createFromDate(Carbon::now()-
>year, $month, $i);
            array_push($attendanceArray, $attribute);
         }
         Attendance::insert($attendanceArray);
      }
      else
      {
```

```
        $month = ($attendance->date->month + 1) > 12 ? 1 :
$attendance->date->month + 1;
        $year = $attendance->date->month == 12 ? $attendance->date-
>year+1 : $attendance->date->year;
        $totalDaysInMonth = Carbon::createFromDate($year, $month,
1)->daysInMonth;

        $attendanceArray = [];
        for($i=1;$i<=$totalDaysInMonth;$i++)
        {
           $attribute['schedule_id'] = 1;
           $attribute['user_id'] = $user_id;
           $attribute['date'] = Carbon::createFromDate($year, $month,
$i);
           array_push($attendanceArray, $attribute);
        }
        Attendance::insert($attendanceArray);
     }

         return redirect()->back()->withMessage('Attendance
generated');
   }

   public function index(Request $request, $user_id)
   {
      if(!$request->date)
      {
         return redirect()->back()->with('message', "Date required");
      }

      $date = Carbon::createFromFormat("Y-m-d", $request->date."-1");
      $attendances = Attendance::where('user_id', $user_id)
                        ->with('schedule')
                        ->whereMonth('date', '=', $date->month)
                        ->whereYear('date', '=', $date->year)
                        ->get();

      if(!$attendances or !count($attendances))
      {
```

```
            return redirect()->back()->with('message', "Sorry, no attendances
in ".$date->format('M Y').". Please generate first");
        }

        $user = User::with(['salaryRelation'=>function($query) use($date){
                    $query->whereMonth('date', $date->month);
                }])->findOrFail($user_id);

        $schedules = Schedule::get();


        $data['title'] = "User Attendance";
        $data['date'] = $date;
        $data['attendances'] = $attendances;
        $data['user'] = $user;
        $data['schedules'] = $schedules;
        return view('attendance.index',$data);
    }

    public function changeSchedule(Request $request, $id, $schedule_id)
    {
        Attendance::where('id',$id)-
>update(['schedule_id'=>$schedule_id]);

        return redirect()->route("attendance-user-index",[$request-
>user_id,$request->date]);
    }


    public function home(Request $request)
    {
        if(!$request->date)
        {
            $today = Carbon::now();
        }
        else
        {
            //$today = Carbon::createFromFormat("Y-m-d",$request->date)-
>hour($request->hour)->minute($request->minute);
            $today = Carbon::now();
```

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

        if($request->has('checkin'))
        {
            $attendance = Attendance::where('user_id', $request->user_id)-
>whereDate('date','=', $today->format('Y-m-d'))->with('schedule')-
>first();
            if(!$attendance)
            {
                return redirect()->back()->withMessage('Sorry, no attendance
record in database. Please contact your administrator.');
            }
            elseif($attendance->schedule_id == SCHEDULE_OFF)
            {
                return redirect()->back()->withMessage('Sorry, today is your
free day');
            }

            $todayTemp = clone $today;
            $startCheckInAt = $attendance->schedule-
>createStartCheckInAt($today);
            $today = clone $todayTemp;
            $stopCheckInAt = $attendance->schedule-
>createStopCheckInAt($today);
            $today = clone $todayTemp;



            if($attendance->check_in)
            {
                return redirect()->back()->withMessage('Sorry, you already
check in at '.$attendance->check_in->format('H:i:s'));
            }
            elseif ($today->lt($startCheckInAt))
            {
                return redirect()->back()->withMessage('Sorry, you may
checkin at '.$startCheckInAt->format('H:i'));
            }
            elseif ($today->gt($stopCheckInAt))
```

```
        {
            $attendance->check_in = $today;
            $attendance->is_late = 1;
            $attendance->save();

            return redirect()->back()->withMessage('Thank you for
coming but you just came late,
                you check in at. '.$today);
        }


        $attendance->check_in = $today;
        $attendance->save();

        return redirect()->back()->withMessage("Thank you, have fun
work for today");
    }
    elseif($request->has('checkout'))
    {
        $attendance = Attendance::where('user_id', $request->user_id)-
>whereDate('date',"=", $today->format('Y-m-d'))->first();

        $todayTemp = clone $today;
        $startCheckOutAt = $attendance->schedule-
>createStartCheckOutAt($today);
        $today = clone $todayTemp;
        if(!$attendance)
        {
            return redirect()->back()->withMessage('Sorry, no attendance
record in database. Please contact your administrator.');
        }
        elseif($attendance->check_out)
        {
            return redirect()->back()->withMessage('Sorry, you already
check out at '.$attendance->check_out);
        }
        elseif (!$attendance->check_in)
        {
            return redirect()->back()->withMessage('Sorry, you need to
check_in first ');
```

```
        }
        elseif ($today->lt($startCheckOutAt))
        {
            return redirect()->back()->withMessage('Sorry, you may
checkout at '.$startCheckOutAt->format('H:i'));
        }

        $attendance->totalwork_hours = $attendance->check_in-
>diffInMinutes($today, false);
        $attendance->check_out = $today;
        $attendance->save();

        return redirect()->back()->withMessage("Thank you for today.");
    }
    elseif($request->has('startbreak'))
    {
        $attendance = Attendance::where('user_id', $request->user_id)-
>whereDate('date',"=", $today->format('Y-m-d'))-
>with('schedule','breaktime')->first();

        if(!$attendance)
        {
            return redirect()->back()->withMessage('Sorry, no attendance
record in database. Please contact your administrator.');
        }
        elseif($attendance->check_out)
        {
            return redirect()->back()->withMessage('Sorry, you already
check out at '.$attendance->check_out->format("H:i:s"));
        }
        elseif (!$attendance->check_in)
        {
            return redirect()->back()->withMessage('Sorry, you need to
checkin first ');
        }
        elseif (count($attendance->breaktime) > 2)
        {
            return redirect()->back()->withMessage('Sorry, you already
got three times breaktime for today.');
        }
```

```
        elseif (count($attendance->breaktime) and $attendance-
>breaktime->sum('duration') > $attendance->schedule-
>breaktime_duration)
        {
            return redirect()->back()->withMessage('Sorry, your
breaktime was rearched maximum breaktime quota');
        }

        foreach ($attendance->breaktime as $breaktime)
        {
            if(!$breaktime->stop_break_at)
            {
                return redirect()->back()->withMessage('Sorry, you have to
stop previous breaktime.');
            }
        }

        $breaktime = new Breaktime();
        $breaktime->attendance_id = $attendance->id;
        $breaktime->start_break_at = $today->format("H:i:s");
        $breaktime->save();

        return redirect()->back()->withMessage("Let's rest in a while.");
    }
    elseif($request->has('stopbreak'))
    {
        $attendance = Attendance::where('user_id', $request->user_id)-
>whereDate('date',"=", $today->format('Y-m-d'))-
>with('schedule','breaktime')->first();

        if(!$attendance)
        {
            return redirect()->back()->withMessage('Sorry, no attendance
record in database. Please contact your administrator.');
        }
        elseif($attendance->check_out)
        {
            return redirect()->back()->withMessage('Sorry, you already
check out at '.$attendance->check_out->format("H:i:s"));
        }
```

```
        elseif (!$attendance->check_in)
        {
            return redirect()->back()->withMessage('Sorry, you need to
checkin first ');
        }
        elseif ($attendance->breaktime->last() and !$attendance-
>breaktime->last()->start_break_at)
        {
            return redirect()->back()->withMessage('Sorry, you already
stop breaktime at '.$attendance->date->format('d M Y'));
        }
        elseif ($attendance->breaktime->last() and $attendance-
>breaktime->last()->stop_break_at)
        {
            return redirect()->back()->withMessage('Sorry, breaktime not
found. You have to start breaktime first.');
        }

        $breaktime = $attendance->breaktime->last();
        $breaktime->stop_break_at = $today->format("H:i:s");
        $breaktime->duration = Carbon::createFromFormat("Y-m-d
H:i:s",$today->format("Y-m-d").$breaktime->start_break_at)-
>diffInMinutes($today, false);
        $breaktime->save();

        return redirect()->back()->withMessage("Let's work again.");
    }
    elseif($request->has('reportattendance'))
    {
        $user = User::where('id',$request->user_id)->first();
        $dateattendance =  Carbon::createFromFormat("Y-m",$request-
>dateattendance);
        $attendances = Attendance::where('user_id',$user->id)-
>whereMonth('date',$dateattendance->month)-
>whereYear('date',$dateattendance->year)->get();
        $data['attendances'] = $attendances;
        $data['user'] = $user;

        return view('attendance.report',$data);
    }
```

```
        elseif($request->has('reportbreaktime'))
        {
            $user = User::where('id',$request->user_id)->first();
            $datebreaktime =  Carbon::createFromFormat("Y-m",$request-
>datebreaktime);
            $attendances = Attendance::where('user_id',$user->id)
                                ->whereMonth('date',$datebreaktime->month)
                                ->whereYear('date',$datebreaktime->year)
                                ->with('breaktime')
                                ->get();

            //dd($attendances);

            $data['attendances'] = $attendances;
            $data['date'] = $datebreaktime;
            $data['user'] = $user;

            return view('breaktime.report',$data);

        }


    }

    public function store()
    {
        # code...
    }

    public function delete()
    {
        # code...
    }
    public function update()
    {
        # code...
    }
    public function edit()
    {
        # code...
```

```
    }

    public function indexBreaktime(Request $request, $id)
    {
        if(!$request->date)
        {
            return redirect()->back()->with('message', "Date required");
        }

        $date = Carbon::createFromFormat("Y-m-d", $request->date);
        $breaktimes = Breaktime::where('attendance_id', $id)->get();
        $schedules = Attendance::where('id', $id)->get();

        if(!$breaktimes or !count($breaktimes))
        {
            return redirect()->back()->with('message', "Sorry, no breaktime
at ".$date->format('d M Y'));
        }

        $data['title'] = "User Breaktime";
        $data['date'] = $date;
        $data['breaktimes'] = $breaktimes;
        $data['total'] = $breaktimes->sum('duration');
        $data['schedules'] = $schedules;
        //dd($data);
        return view('breaktime.index',$data);
    }

    public function countSalary(Request $request, $user_id)
    {
        if(!$request->date)
        {
            return redirect()->back()->with('message', "Date required");
        }
        $user = User::with('position')->findOrFail($user_id);

        $date = Carbon::createFromFormat("Y-m-d", $request->date);
        $salary = Salary::where('user_id', $user_id)
                    ->whereMonth('date', '=', $date->month)
                    ->whereYear('date', '=', $date->year)
```

```
                ->count();
        if($salary)
        {
            return redirect()->back()->with('message', "Sorry, salary already
counted for this month");
        }

        $attendances = Attendance::where('user_id', $user_id)
                            ->with('schedule')
                            ->whereMonth('date', '=', $date->month)
                            ->whereYear('date', '=', $date->year)
                            ->get();

        $totalHoursMonthly = $attendances->sum('totalwork_hours')/60;

        if($totalHoursMonthly > 208)
        {
            $totalOvertimeSalary = 0;
            foreach ($attendances as $key => $attendance)
            {
                $totalOvertimeSalary = ($attendance-
>totalwork_hours*$user->position->overtime_salary) +
$totalOvertimeSalary;
            }
        }
        else
        {
            $totalOvertimeSalary = 0;
        }

        $salary = new Salary();
        $salary->user_id = $user->id;
        $salary->overtime_salary = $totalOvertimeSalary;
        $salary->salary = $user->salary;
        $salary->allowance = $user->allowance;
        $salary->date = $date;
        $salary->save();

        return redirect()->back()->withMessage("Salary counted
successfully..");
```

```
    }


}
```

3. homecontroller.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Company;

class HomeController extends Controller
{
    /**
     * Create a new controller instance.
     *
     * @return void
     */
    public function __construct()
    {
        // prevent guest(not login yet) to access this controller
        // $this->middleware('auth');
    }

    /**
     * Show the application dashboard.
     *
     * @return \Illuminate\Http\Response
     */
    public function dashboard()
    {
        // dd(Company::get());
        // dd(\Hash::make('1234567'));

        // make a query to get all data in company table
        //$company = Company::get();
```

```
        // call user's company
        // dd($company->user);
        // dd(ROLE_ADMIN);

        // $data['companyss'] = $company;
        // $data['companys'] = $company;
        //dd($data);
        return view('dashboard');
    }

    public function index()
    {
        return view('home');
    }

    public function report(){
        return  view('report');
    }
}
```

4. Schedulecontroller.php

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use App\Position;
use App\BrandName;
use App\Schedule;
use App\Company;
use App\User;
use App\Attendance;
use Carbon\Carbon;

class ScheduleController extends Controller
{

            public function index(Request $request, $user_id)
```

```
    {
        if(!$request->date)
        {
            return redirect()->back()->with('message', "Date
required");
        }
        $date = Carbon::createFromFormat("Y-m-d", $request->date."-
1");
      $attendances = Attendance::where('user_id', $user_id)
                                                                    -
>with('user', 'schedule')
                                                                    -
>whereMonth('date', '=', $date->month)

        ->whereYear('date', '=', $date->year)
                                                                    ->get();
      $schedules = Schedule::get();


      $data['title'] = "List Schedule";
      $data['date'] = $date;
      $data['attendances'] = $attendances;
      $data['schedules'] = $schedules;
      return view('attendance.index',$data);
    }

    public function changeSchedule($date, $schedule_id)
    {
        dd("asd");
    }
}
```

5. User controller.php

```
<?php

namespace App\Http\Controllers;
```

```
use Illuminate\Http\Request;
use Illuminate\Support\Facades\DB;
use Illuminate\Pagination\Paginator;
use App\Position;
use App\BrandName;
use App\Department;
use App\Company;
use App\User;

class UserController extends Controller
{
   public function create()
   {
      //dd(\Hash::make('1234567'));
      $position = Position::get();
      $brandNames = BrandName::get();
      $departments = Department::get();
      $companys = Company::get();

            $data['title'] = "Create User";
      $data['positions'] = $position;
      $data['brandNames'] = $brandNames;
      $data['departments'] = $departments;
      $data['companys'] = $companys;
            return view('user.create', $data);
   }

   public function index()
   {
      $users = User::paginate(3);


            $data['title'] = "List User";
      $data['users'] = $users;
            // return view('user.index',['users'=>$users]);
      return view('user.index', $data);
   }

   public function store(Request $request)
   {
```

```
        // dd($request->all());

        $user = new User();
        $user->id = $request->id;
        $user->role_id = $request->role_id;
        $user->name = $request->name;
        $user->gender = $request->gender;
        $user->birth_date = $request->birth_date;
        $user->birth_city = $request->birth_city;
        $user->address = $request->address;
        $user->phone = $request->phone;
        $user->salary = $request->salary;
        $user->password = $request->password;
        $user->position_id= $request->position_id;
        $user->brand_name_id = $request->brand_name_id;
        $user->department_id = $request->department_id;
        $user->company_id = $request->company_id;
        $user->save();

        $positions = Position::get();
        $brandNames = BrandName::get();
        $departments = Department::get();
        $companys = Company::get();

        $data['title'] = "Create User";
        $data['positions']=$positions;
        $data['brandNames'] = $brandNames;
        $data['departments'] = $departments;
        $data['companys'] = $companys;
        return view('user.create', $data);
    }

    public function delete($id)
    {
        $users = User::findOrFail($id);
        $users->delete();
        // return redirect()->route('blog.index')->with('alert-success','Data
Hasbeen Saved!');
        // dd(redirect()->route('user-index')->with('message','Data terhapus
bro!!'))
```

```
        return redirect()->route('user-index')->with('message','Your data
has been deleted');
    }

    public function edit($id)
    {
        $user = User::findOrFail($id);
        $position = Position::get();
        $brandNames = BrandName::get();
        $departments = Department::get();
        $companys = Company::get();

        $data['title'] = "Edit User";
        $data['positions'] = $position;
        $data['user'] = $user;
        $data['brandNames'] = $brandNames;
        $data['departments'] = $departments;
        $data['companys'] = $companys;
        return view('user.edit', $data);
    }

    public function update(Request $request, $id)
    {
        $user = User::findOrFail($id);
        $user->id = $request->id;
        $user->role_id = $request->role_id;
        $user->name = $request->name;
        $user->gender = $request->gender;
        $user->birth_date = $request->birth_date;
        $user->birth_city = $request->birth_city;
        $user->address = $request->address;
        $user->phone = $request->phone;
        $user->salary = $request->salary;
        $user->password = $request->password;
        $user->position_id= $request->position_id;
        $user->brand_name_id = $request->brand_name_id;
        $user->department_id = $request->department_id;
        $user->company_id = $request->company_id;
        $user->save();
        $users = DB::table('users')->paginate(10);
```

```
        $data['title'] = "List User";
        $data['users'] = $users;
        // return view('user.index',['users'=>$users]);
        return redirect()->route('user-index')->with('message','Your data
has been updated');
    }

}
```

6.Attendance.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Attendance extends Model
{
    protected $table = "attendance";

    protected $dates = ['date','check_in','check_out'];

    public function breaktime()
    {
        return $this->hasMany('App\Breaktime');
    }
    public function schedule()
    {
            return $this->belongsTo('App\Schedule');
    }
    public function user()
    {
            return $this->belongsTo('App\User');
    }
}
```

7.Brandname.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class BrandName extends Model
{
   protected $table = "brand_name";



   public function user()
   {
      return $this->hasMany('App\User');
   }
}
```

8.Breaktime.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Breaktime extends Model
{
   protected $table = "breaktime";



   public function attendance()
   {
      return $this->belongsTo('App\Attendance');
   }
}
```

9.Company.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Company extends Model
{
   protected $table = "company";



   public function user()
   {
      return $this->hasMany('App\User');
   }
}
```

10.Constant.php

```
<?php
define('ROLE_ADMIN',1);
define('ROLE_USER',2);


define('SCHEDULE_PAGI',1);
define('SCHEDULE_SIANG',2);
define('SCHEDULE_LEMBUR',3);
define('SCHEDULE_OFF',4);
?>
```

11.Department.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;
```

```
class Department extends Model
{
   protected $table = "department";

   public function User()
   {
          return $this->hasMany('App\User');
   }
}
```

12.Position.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Position extends Model
{
   protected $table = "position";



   public function user()
   {
      return $this->hasMany('App\User');
   }
}
```

13.Sallary.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;
```

```
class Salary extends Model
{
   protected $table = "salary";

   protected $dates = ["date"];

   public function user()
   {
      return $this->belongsTo('App\User');
   }

   public function getTotalSalaryAttribute()
   {
         return (int)$this->salary + (int)$this->allowance + (int)$this-
>overtime_salary;
   }

   public function getTotalSalaryViewAttribute()
   {
         $total =  (int)$this->salary + (int)$this->allowance + (int)$this-
>overtime_salary;
         return "Rp.".number_format($total, 2, ',', '.');
   }

   public function getAllowanceViewAttribute()
   {
         return "Rp.".number_format($this->allowance, 2, ',', '.');
   }

   public function getSalaryViewAttribute()
   {
         return "Rp.".number_format($this->salary, 2, ',', '.');
   }

   public function getOvertimeSalaryViewAttribute()
   {
         return "Rp.".number_format($this->overtime_salary, 2, ',', '.');
   }
}
```

14.Schedule.php

```
<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class Schedule extends Model
{
   protected $table = "schedule";


   public function attendance()
   {
      return $this->hasMany('App\Attendance');
   }

   public function getStartCheckinHourAttribute()
   {
         return explode(":", $this->start_checkin_at)[0];
   }

   public function getStartCheckinMinuteAttribute()
   {
         return explode(":", $this->start_checkin_at)[1];
   }

   public function getStopCheckinHourAttribute()
   {
         return explode(":", $this->stop_checkin_at)[0];
   }

   public function getStopCheckinMinuteAttribute()
   {
         return explode(":", $this->stop_checkin_at)[1];
   }

   public function createStartCheckInAt($date)
   {
```

```
            try
            {
                return $date->hour(explode(":", $this->start_checkin_at)[0])->minute(explode(":", $this->start_checkin_at)[1])->second(explode(":", $this->start_checkin_at)[2]);
            }
            catch (\Exception $e)
            {
                return null;
            }
    }

    public function createStopCheckInAt($date)
    {
            try
            {
                return $date->hour(explode(":", $this->stop_checkin_at)[0])->minute(explode(":", $this->stop_checkin_at)[1])->second(explode(":", $this->stop_checkin_at)[2]);
            }
            catch (\Exception $e)
            {
                return null;
            }
    }

    public function createStartCheckOutAt($date)
    {
            try
            {
                return $date->hour(explode(":", $this->start_checkout_at)[0])->minute(explode(":", $this->start_checkout_at)[1])->second(explode(":", $this->start_checkout_at)[2]);
            }
            catch (\Exception $e)
            {
                return null;
```

```
            }
    }

}
```

15.User.php

```
<?php

namespace App;

use Illuminate\Notifications\Notifiable;
use Illuminate\Foundation\Auth\User as Authenticatable;


class User extends Authenticatable
{
    use Notifiable;


    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = [
        'name', 'email', 'password',
    ];

    /**
     * The attributes that should be hidden for arrays.
     *
     * @var array
     */
    protected $hidden = [
        'password', 'remember_token',
    ];
```

```
    protected $dates = [
        'birth_date'
    ];


    public function company()
    {
        return $this->belongsTo('App\Company');
    }

    public function department()
    {
        return $this->belongsTo('App\Department');
    }
    public function brandName()
    {
        return $this->belongsTo('App\BrandName');
    }
    public function position()
    {
        return $this->belongsTo('App\Position');
    }
    public function salaryRelation()
    {
        return $this->hasMany('App\Salary');
    }
    public function attendance()
    {
        return $this->hasMany('App\Attendance');
    }
}
```