**EMS (Modules)**

**Summary**

In this CRM(EMS) :-

* This is the Technical Documentation for Modules, I (Ekta) made for EMS.
* Structure :

->Administrator

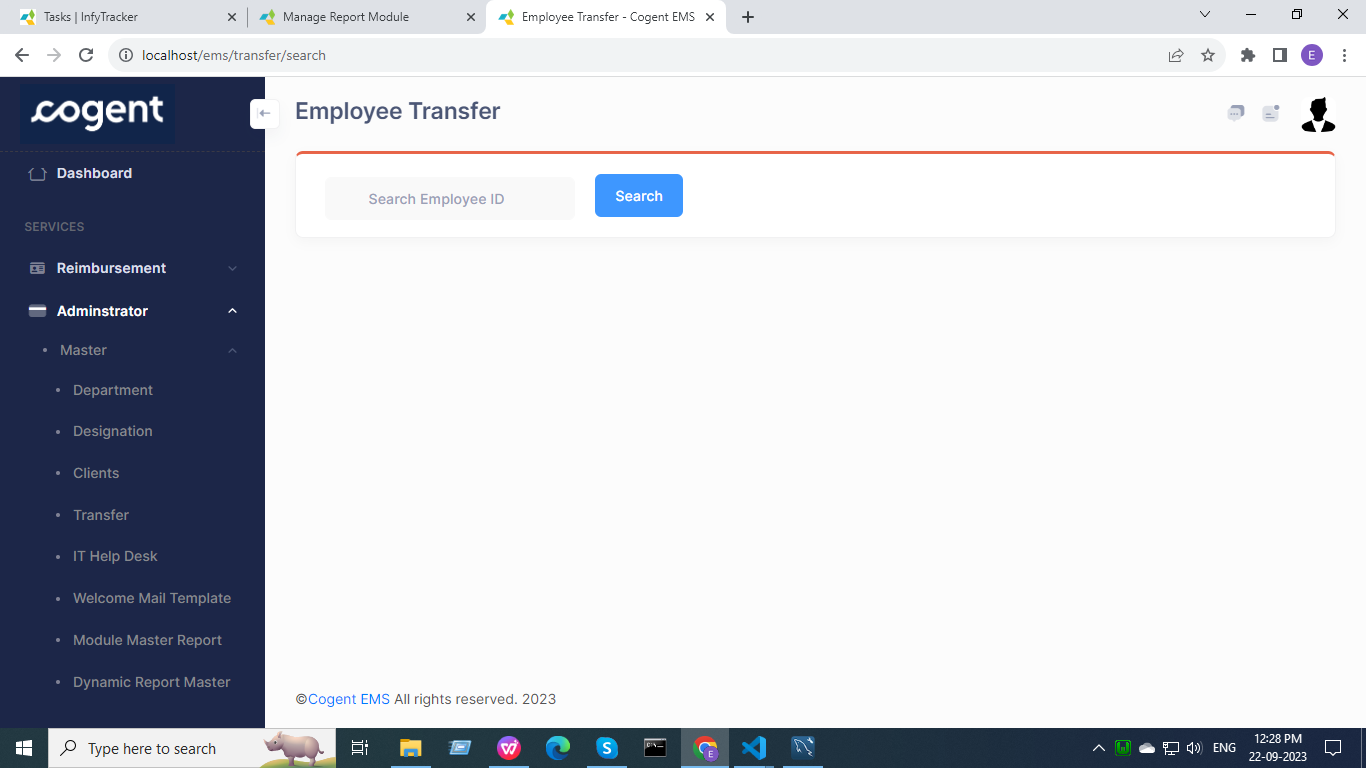
-> Masters

* Transfer (Employee)
* IT Help Desk
* Welcome Mail Template
* Module Master Report
* Dynamic Report Master
* Dynamic Report
* Reference Master
* Reference Report
* Issue Master
* Master Data

**Requirement**

* PHP( Version 8.1 )
* Laravel
* My Sql
* Wamp Server
* VS Code Editor
* Composer

**Transfer (Employee) :-**

****

Routes :-

use App\Http\Controllers\Transfer\TransferController;

/\*For Employee Transfer Routes\*/

Route::any('transfer/search', [TransferController::class, 'search'])->name('transfer.search');

Route::post('/clientname', [TransferController::class, 'clientname'])->name('clientname');

Route::post('/process', [TransferController::class, 'process'])->name('process');

Route::post('/subprocess', [TransferController::class, 'subprocess'])->name('subprocess');

Route::post('/reportTo', [TransferController::class, 'reportTo'])->name('reportTo');

Route::post('save', [TransferController::class, 'save'])->name('save');

Controller :-

namespace App\Http\Controllers\Transfer;

use App\Http\Controllers\Controller;

Blade File :-

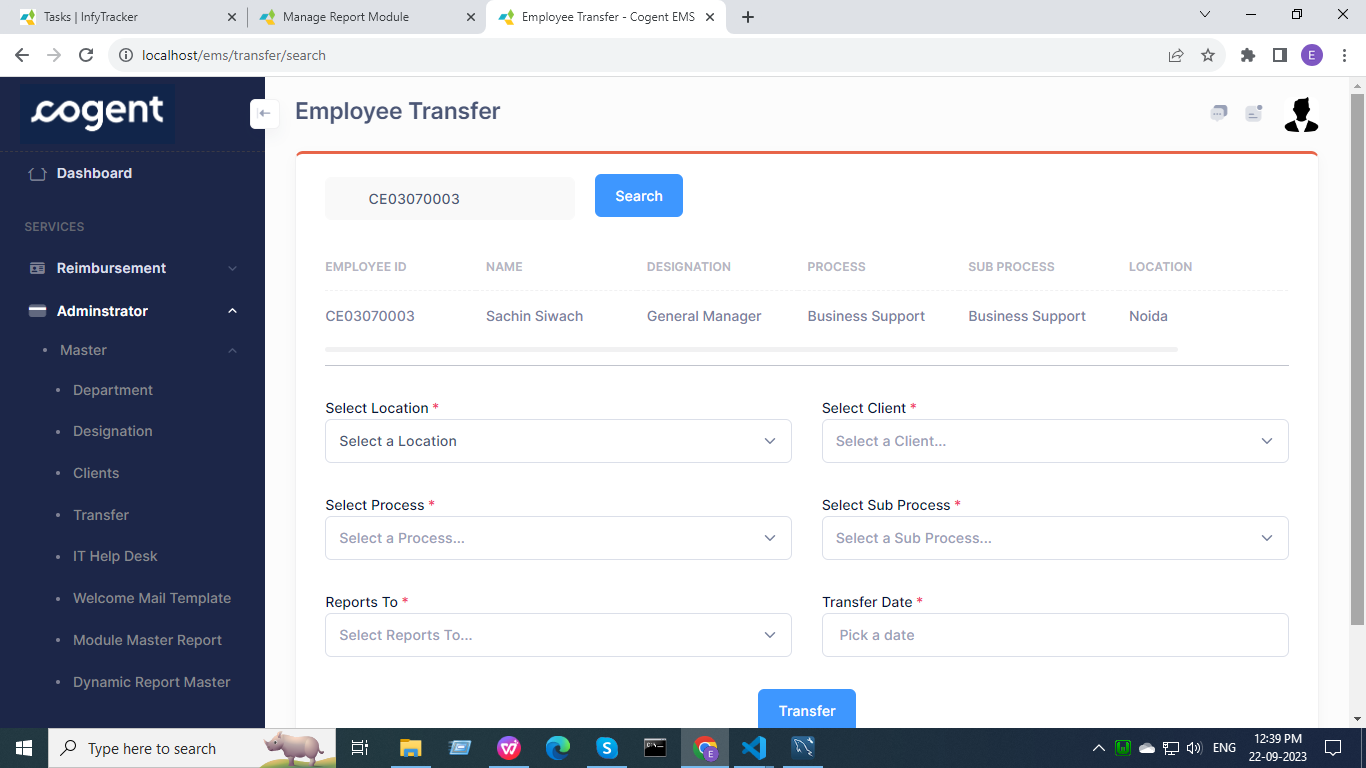
  return view('transfer/transferEmployee');

Models :-

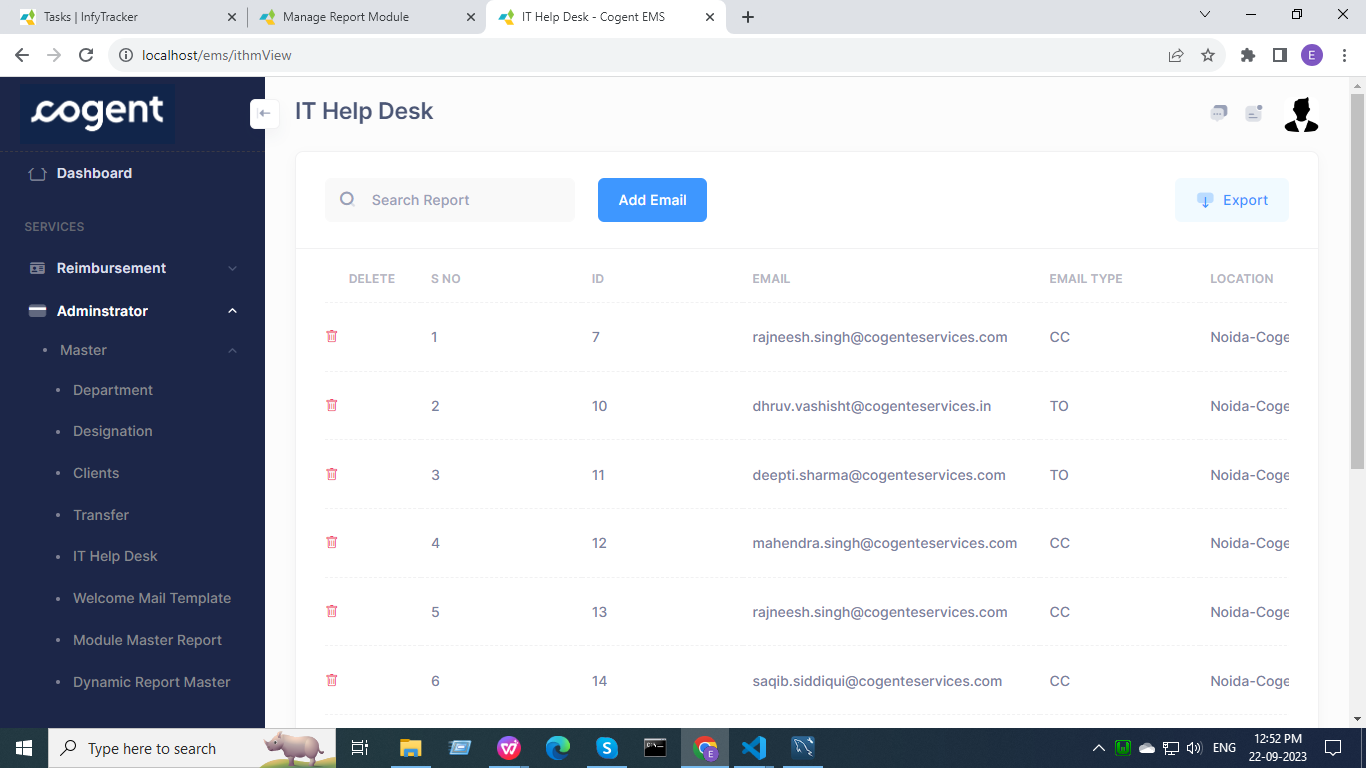
use App\Models\Transfer\EmpDetail;

Function :-

* The user can search for the Employee ID and if the ID is found, data will be fetched.
* Data of the employee ID searched will be displayed on a data-table and under that table, drop-down sections are presented that are dependent on each other along with a date-picker.
* Dependency of the drop-downs is shown through arrows.



**IT Help Desk :-**

****

Routes :-

use App\Http\Controllers\Ithm\IthmController;

/\*For IT Help Desk Routes\*/

Route::get('ithmView', [IthmController::class, 'view'])->name('view');

Route::get('delete-email/{id}', [IthmController::class, 'deleteEmail'])->name('delete-email');

Route::post('saveEmail', [IthmController::class, 'save'])->name('saveEmail');

Controller :-

namespace App\Http\Controllers\Ithm;

use App\Http\Controllers\Controller;

Blade File :-

    return view('ithm/view');

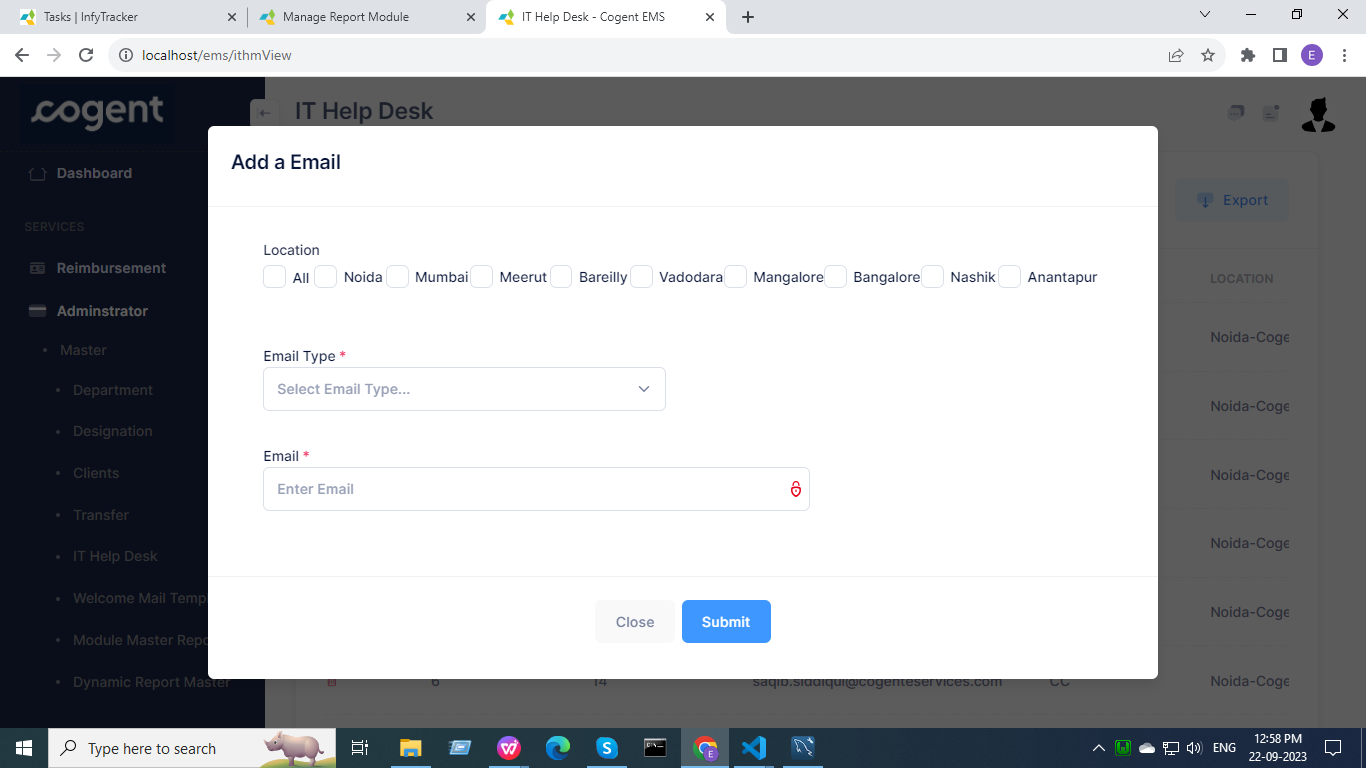
Models :-

use App\Models\Ithm\IthmModel;

use App\Models\Ithm\location\_master;

Function :-

* Data fetched from the database is displayed in a data-table.
* Users can search for a particular entry.
* Users can export the data in Excel or CSV format.
* Users can delete a particular entry from the database.
* Users can add new email details by clicking on ADD EMAIL button



**Welcome Mail Template :-**

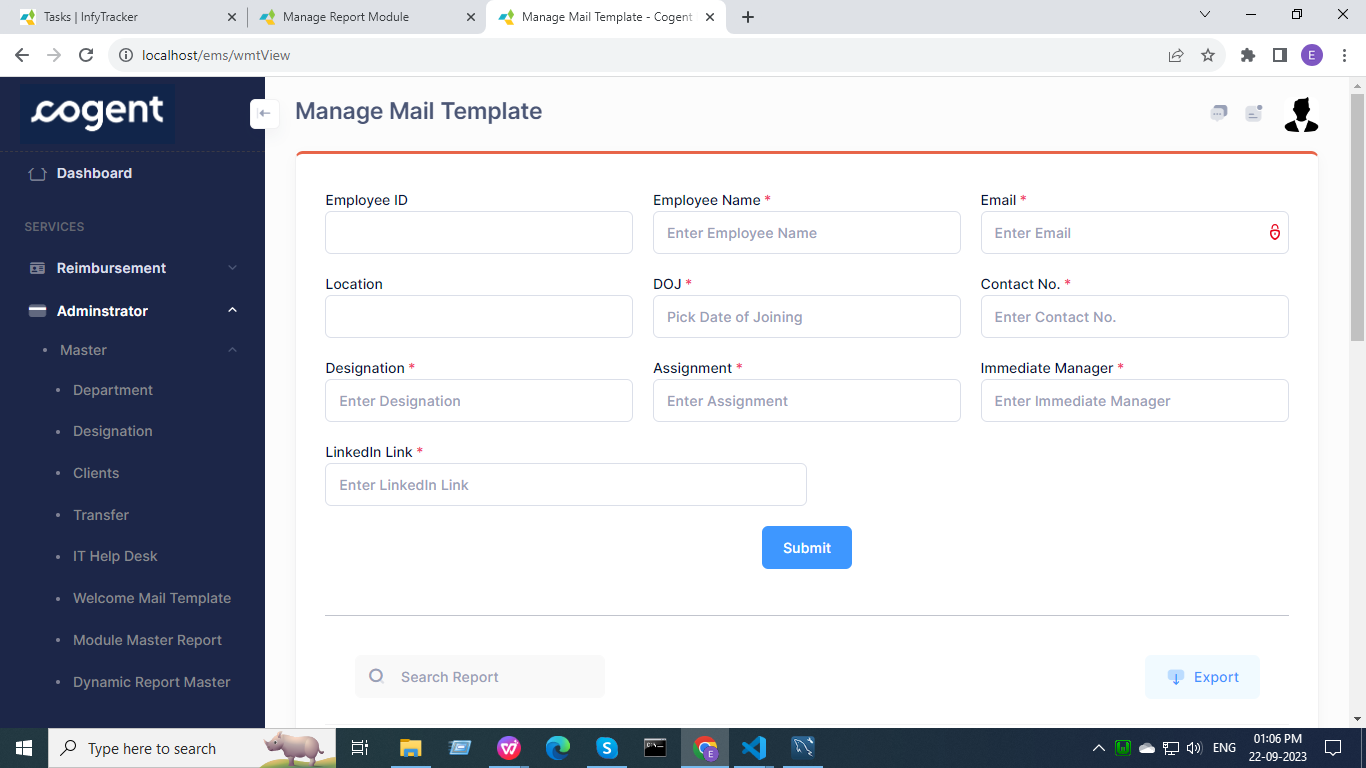
Routes :-

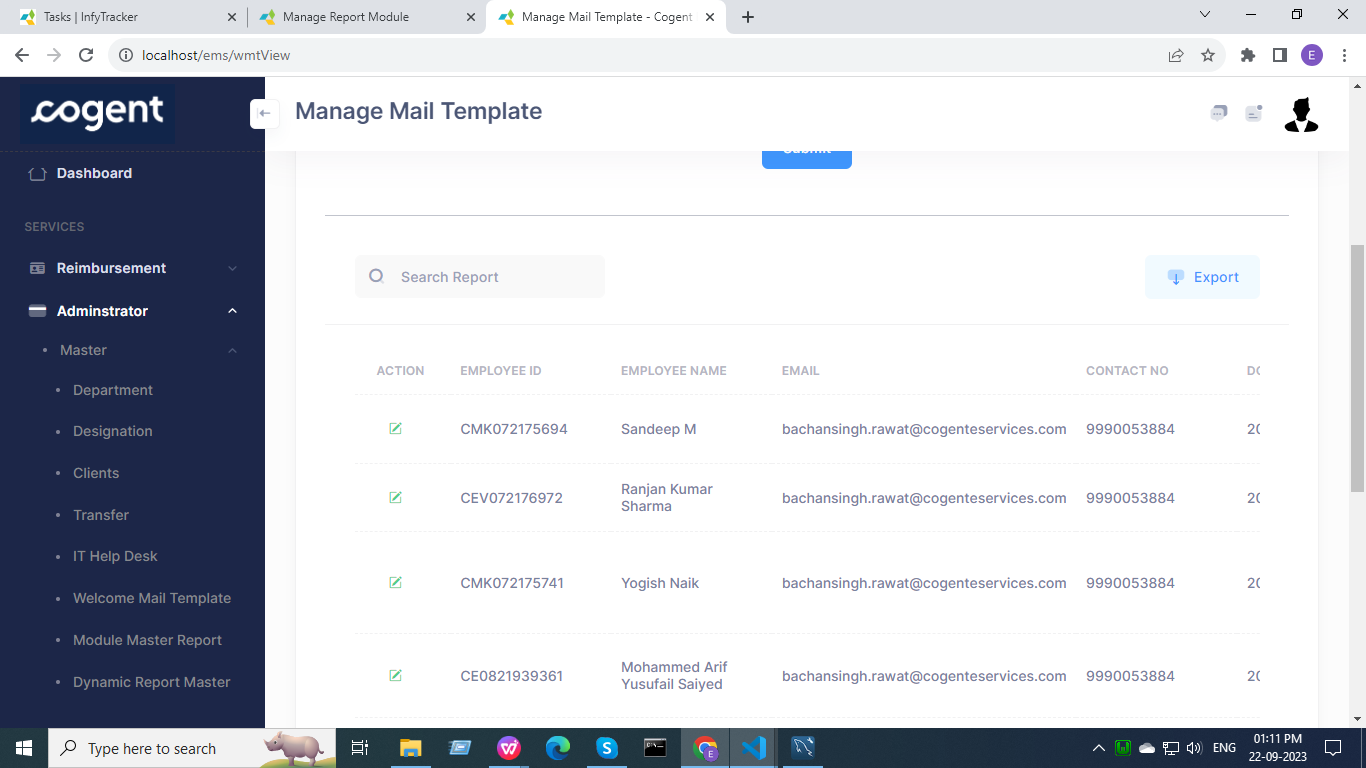
use App\Http\Controllers\Wmt\WmtController;

/\*For Welcome Mail Template Routes\*/

Route::get('wmtView', [WmtController::class, 'view'])->name('wmtView');

Route::post('wmtUpdate', [WmtController::class, 'update'])->name('wmtUpdate');





Controller :-

namespace App\Http\Controllers\Wmt;

use App\Http\Controllers\Controller;

Models :-

use App\Models\wmt\ContactDetail;

use App\Models\Wmt\MailTemplate;

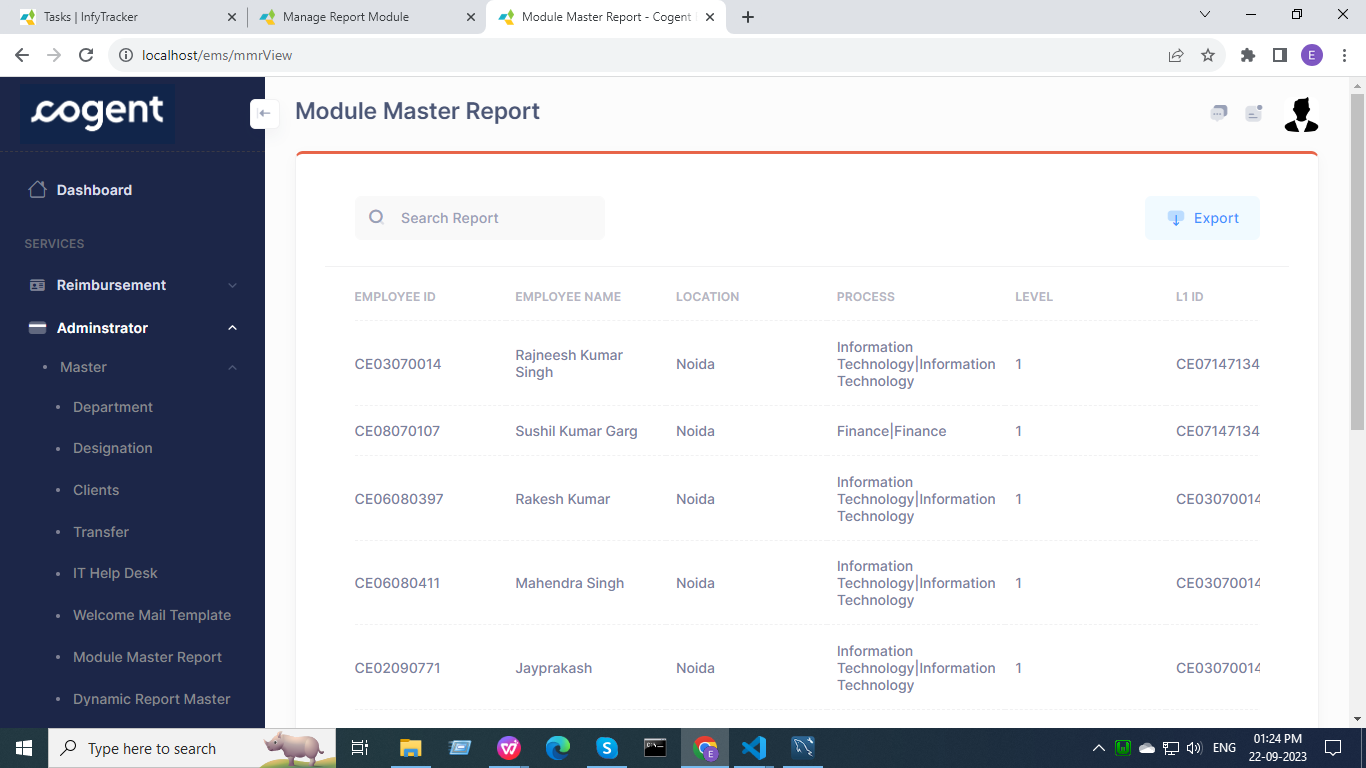
Blade File :-

 return view('wmt/wmtView');

Function :-

* Data fetched from the database is displayed in a data-table.
* Users can search for a particular entry.
* Users can export the data in Excel or CSV format.
* Users can edit a particular entry from the database.
* By clicking on the edit button, the entries of that particular user get auto filled in the input sections above.
* The user can update the details and save the data in a database, data-table will be auto updated with entries.

**Module Master Report :-**

****

Routes :-

use App\Http\Controllers\Mmr\ModuleMasterReport;

/\*For Module Master Report Routes\*/

Route::get('mmrView', [ModuleMasterReport::class, 'view'])->name('mmrView');

Controller :-

namespace App\Http\Controllers\Mmr;

use App\Http\Controllers\Controller;

Blade File :-

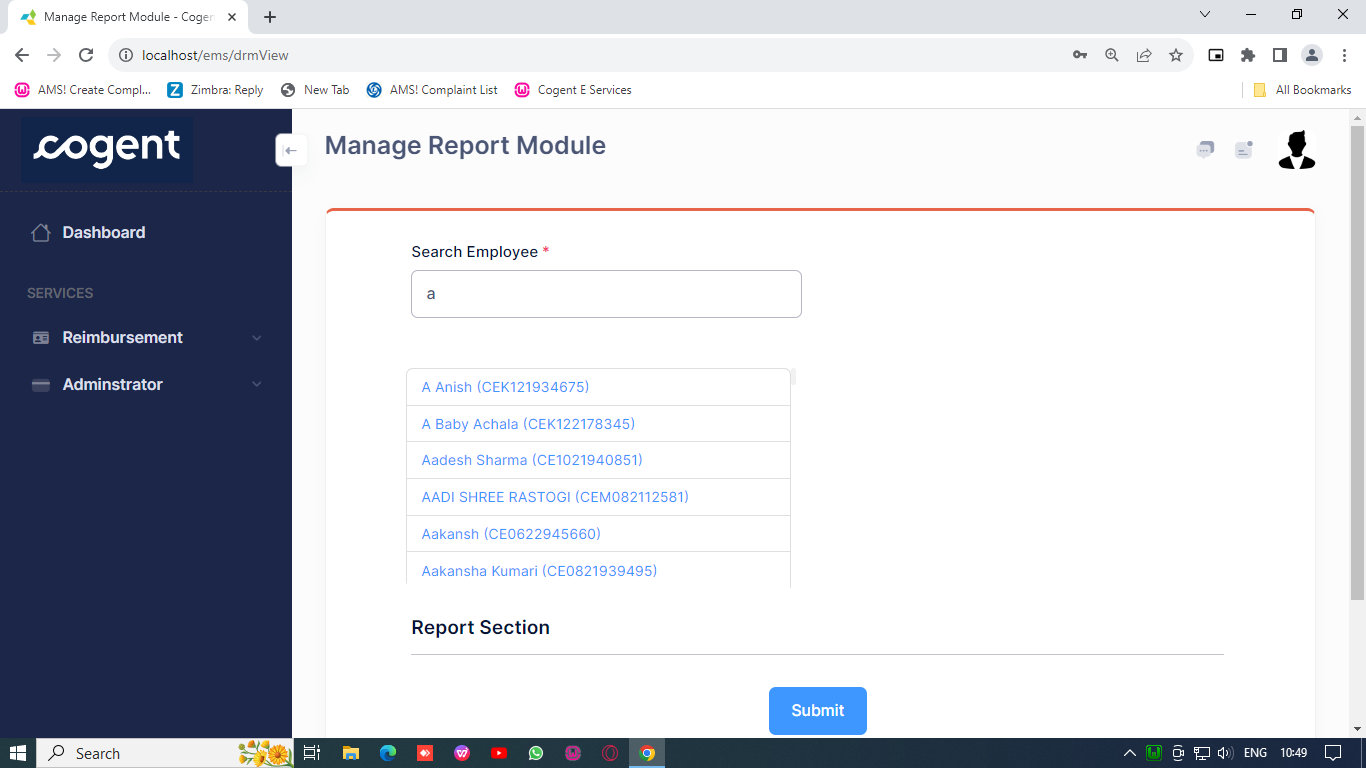
return view('mmr/moduleMasterReport');

Function :-

* Data fetched from the database is displayed in a data-table.
* Users can search for a particular entry.
* Users can export the data in Excel or CSV format.
* The total entries in the database are < 45000, so the page loading time was exceeding the max\_execution\_time of Laravel I.e. 120 (2 min) so we are fetching only limited data-entries to avoid page crashing.

->take(10000)

**Dynamic Report Master :-**



Routes :-

use App\Http\Controllers\Drm\DynamicReportMaster;

/\*For Dynamic Report Master Routes\*/

Route::get('drmView', [DynamicReportMaster::class, 'view'])->name('drmView');

Route::get('get-edit-id/{id}', [DynamicReportMaster::class, 'getEditId'])->name('get-edit-id');

Route::post('drmUpdate', [DynamicReportMaster::class, 'update'])->name('drmUpdate');

Controller :-

namespace App\Http\Controllers\Drm;

use App\Http\Controllers\Controller;

use Illuminate\Http\Request;

use Exception;

use App\Models\Drm\ReportMap;

use Illuminate\Support\Facades\Session;

use Illuminate\Support\Facades\DB;

Model :-

namespace App\Models\Drm;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class ReportMap extends Model

{

    use HasFactory;

    protected $table = "report\_map";

    protected $fillable = ['EmpID', 'reportID','processID', 'CreatedBy'];

    public $timestamps = false;

}

Blade File :-

 return view('drm/dynamicReportMaster', compact('result', 'reports', 'processes'));

Ajax File :-

 $html = response()->json([

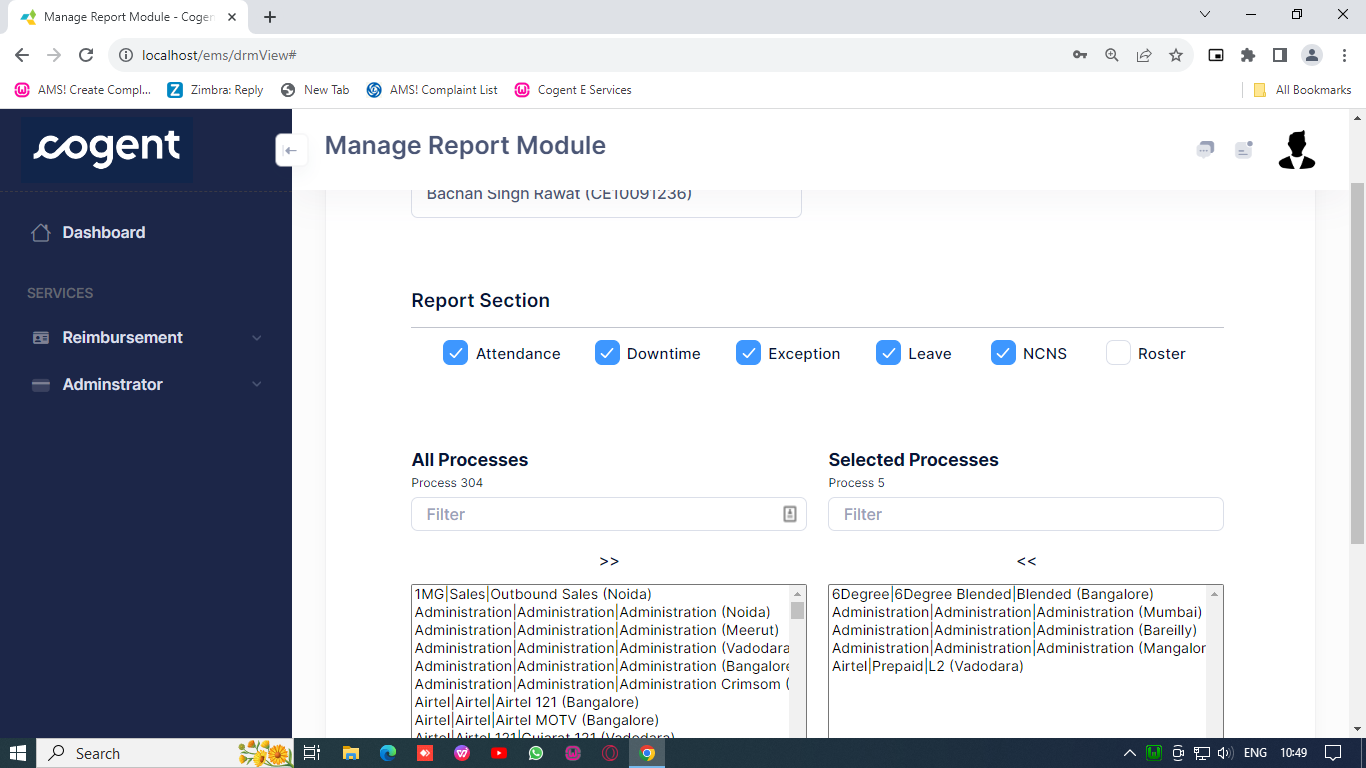
 'reports' => view('ajax.dynamic.report', compact('reportIDs', 'reports'))->render(),

 'process' => view('ajax.dynamic.process', compact('query', 'processes'))->render()

  ]);

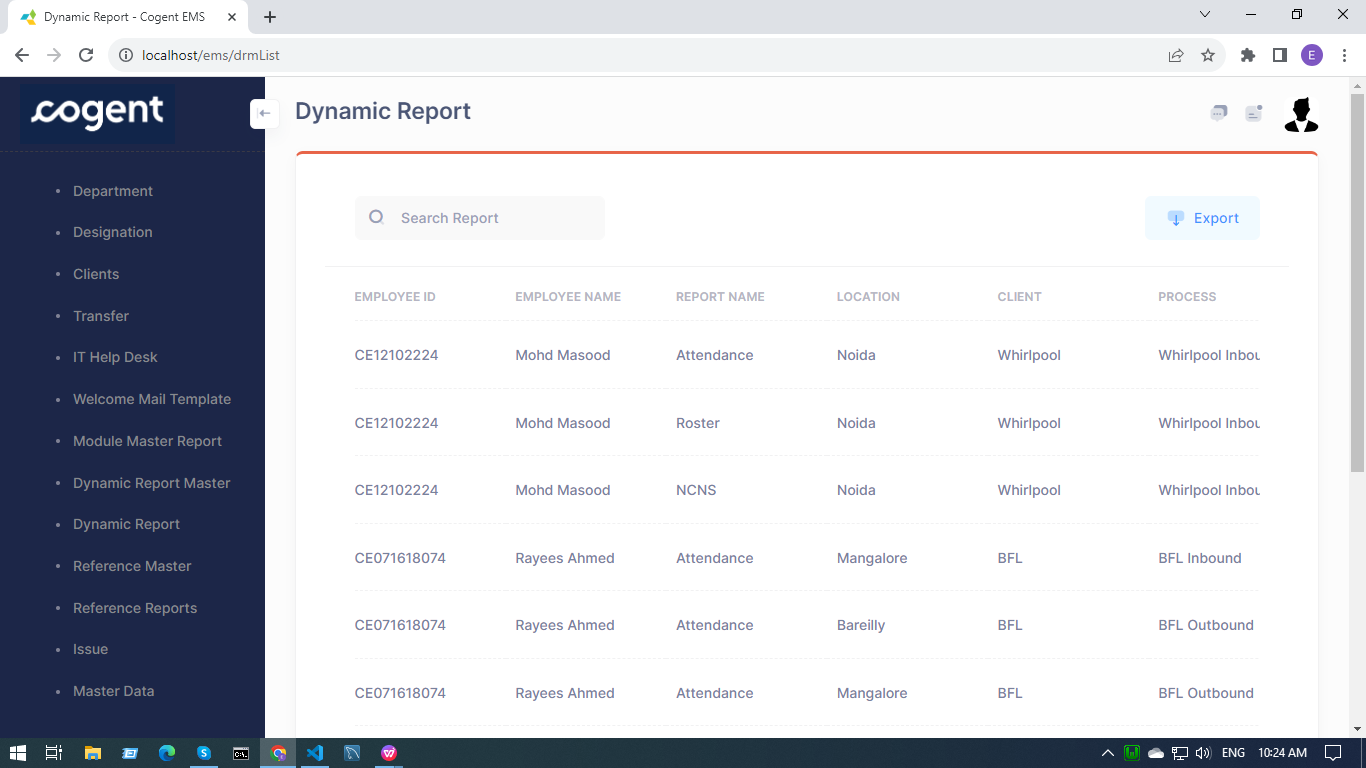
Function :-

* The user will search for an employee to check or enter the report and process data.
* Search employee is mandatory.
* Once the employee is searched, 2 sections appear on the screen that are for : report data and process data.



* The Report section is checkbox type and the Process section is duallistbox type.
* The data in both the sections is being fetched from the database.
* Both the report section and process section can be altered and new data can be saved in the database by clicking the submit button.
* The ‘>>’ and ‘<<’ buttons are used to transfer all the processes to the other hand side.

**Dynamic Report :-**



Routes :-

use App\Http\Controllers\Drm\DynamicReportMaster;

/\*For Dynamic Report Routes\*/

Route::get('drmList', [DynamicReportMaster::class, 'list'])->name('drmList');

Controller :-

namespace App\Http\Controllers\Drm;

use App\Http\Controllers\Controller;

use Illuminate\Http\Request;

use Exception;

use App\Models\Drm\ReportMap;

use Illuminate\Support\Facades\Session;

use Illuminate\Support\Facades\DB;

Model :-

namespace App\Models\Drm;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class ReportMap extends Model

{

    use HasFactory;

    protected $table = "report\_map";

    protected $fillable = ['EmpID', 'reportID','processID', 'CreatedBy'];

    public $timestamps = false;

}

Blade File :-

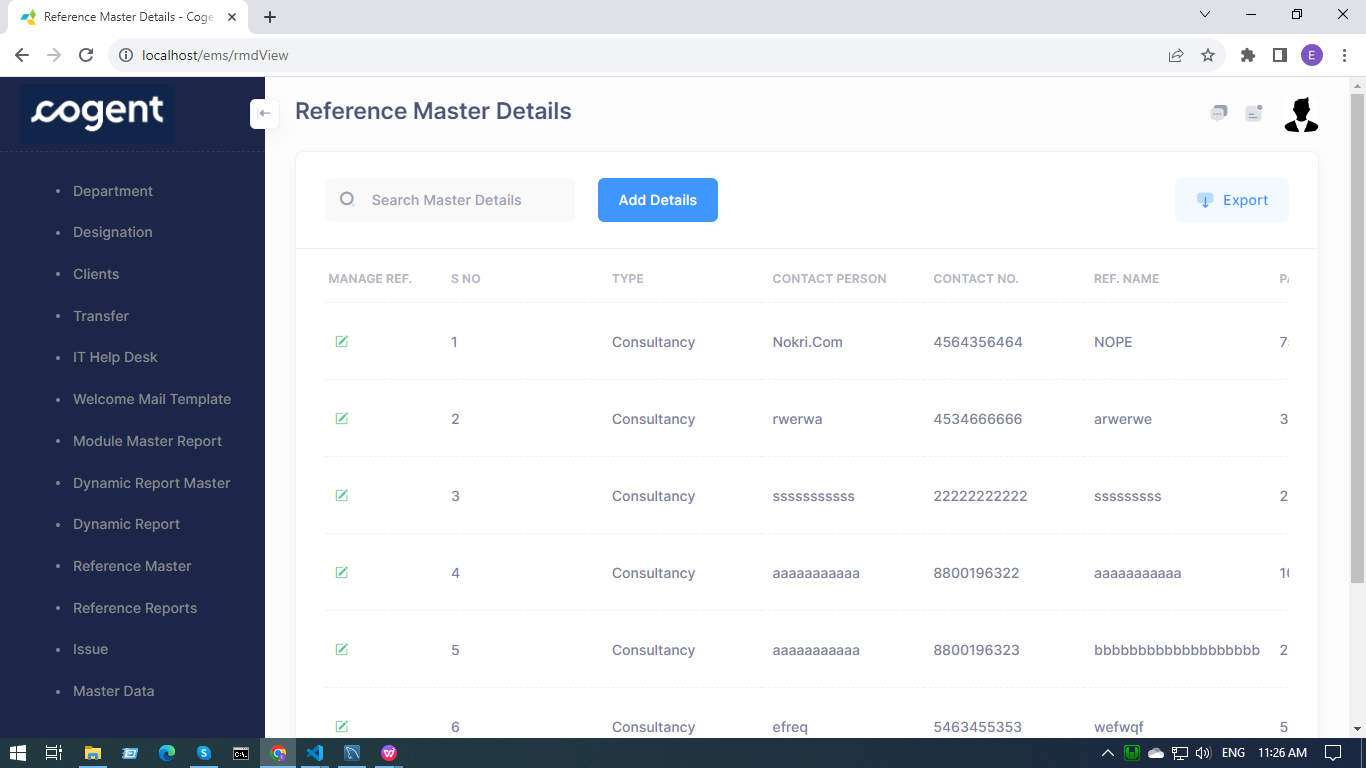
return view('drm/dynamicReportList', compact('reportData'));

Function :-

* Data fetched from the database is displayed in a data-table.
* Users can search for a particular entry.
* Users can export the data in Excel or CSV format.
* The total entries in the database are < 45000, so the page loading time was exceeding the max\_execution\_time of Laravel I.e. 120 (2 min) so we are fetching only limited data-entries to avoid page crashing.

->take(10000)

**Reference Master :-**



Routes :-

use App\Http\Controllers\Rmd\ReferenceMaster;

/\*For Reference Master Details Routes\*/

Route::get('rmdView', [ReferenceMaster::class, 'view'])->name('rmdView');

Route::any('rmdUpdate', [ReferenceMaster::class, 'update'])->name('rmdUpdate');

Route::post('rmdSave', [ReferenceMaster::class, 'save'])->name('rmdSave');

Route::post('rmdAdd', [ReferenceMaster::class, 'add'])->name('rmdAdd');

Controller :-

namespace App\Http\Controllers\Rmd;

use App\Http\Controllers\Controller;

use Illuminate\Support\Facades\Session;

use App\Models\Rmd\RefMaster;

use Illuminate\Http\Request;

use Exception;

use Illuminate\Support\Facades\DB;

Model :-

namespace App\Models\Rmd;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class RefMaster extends Model

{

    use HasFactory;

    protected $table = "ref\_master";

    protected $fillable = ['Type', 'ContactPerson', 'ContactNo', 'RefName', 'Payout', 'Status', 'ModifiedBy', 'ModifiedOn'];

    public $timestamps = false;

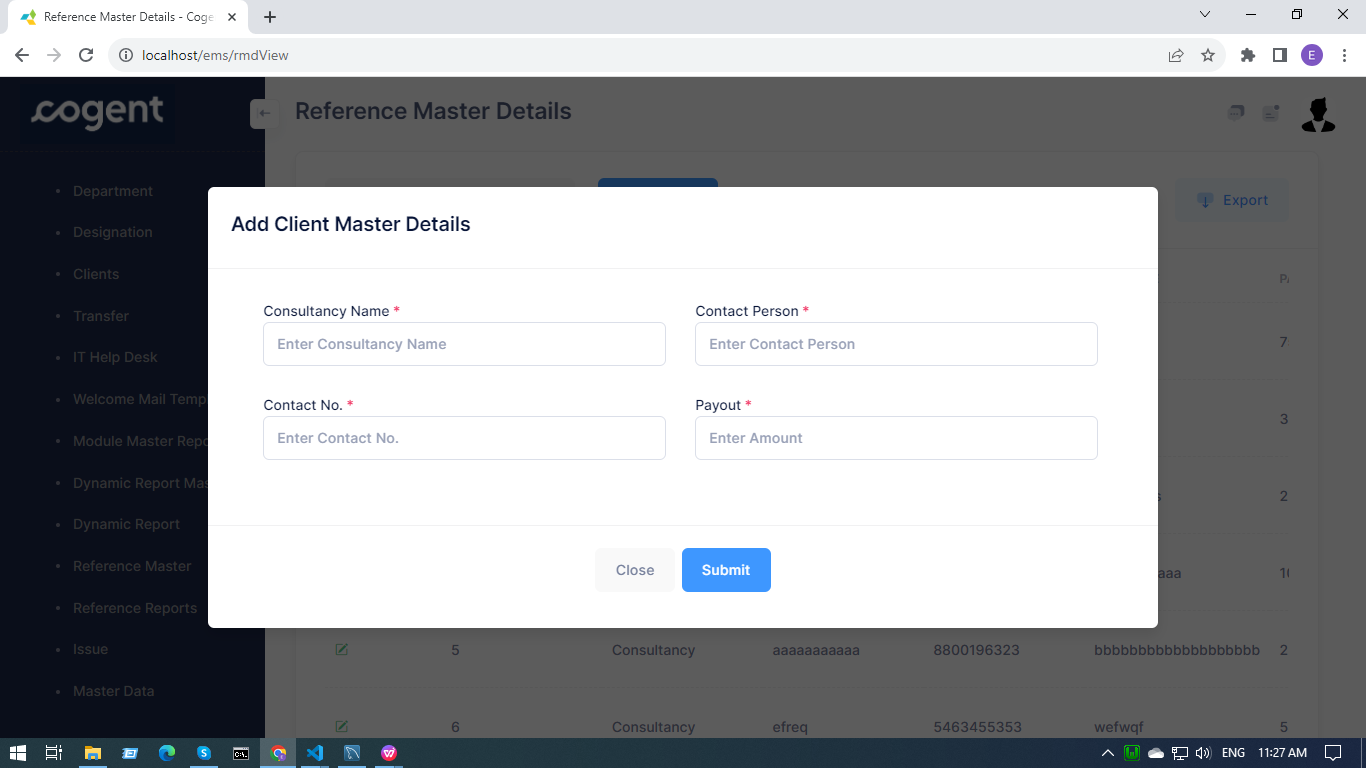
}

Blade File :-

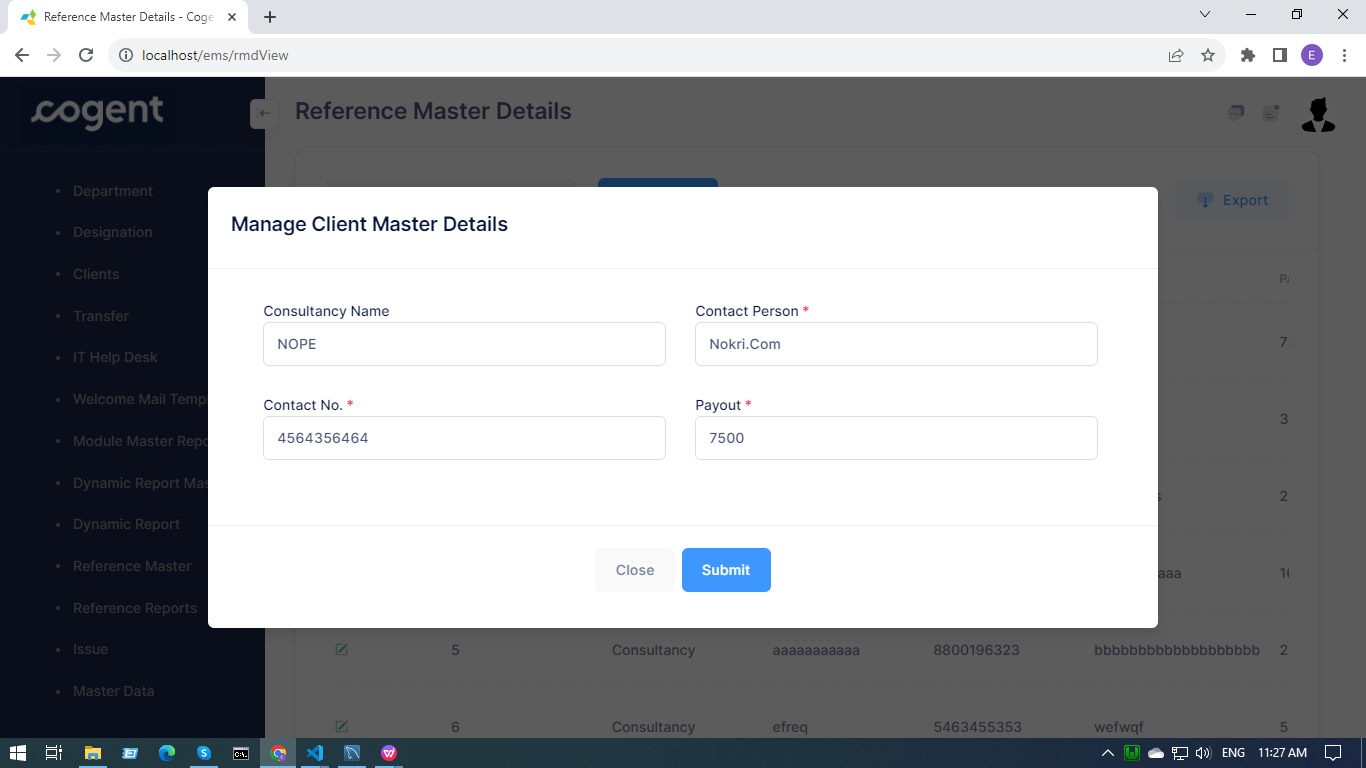
 return view('rmd/rmdView', compact('refData'));

Function :-

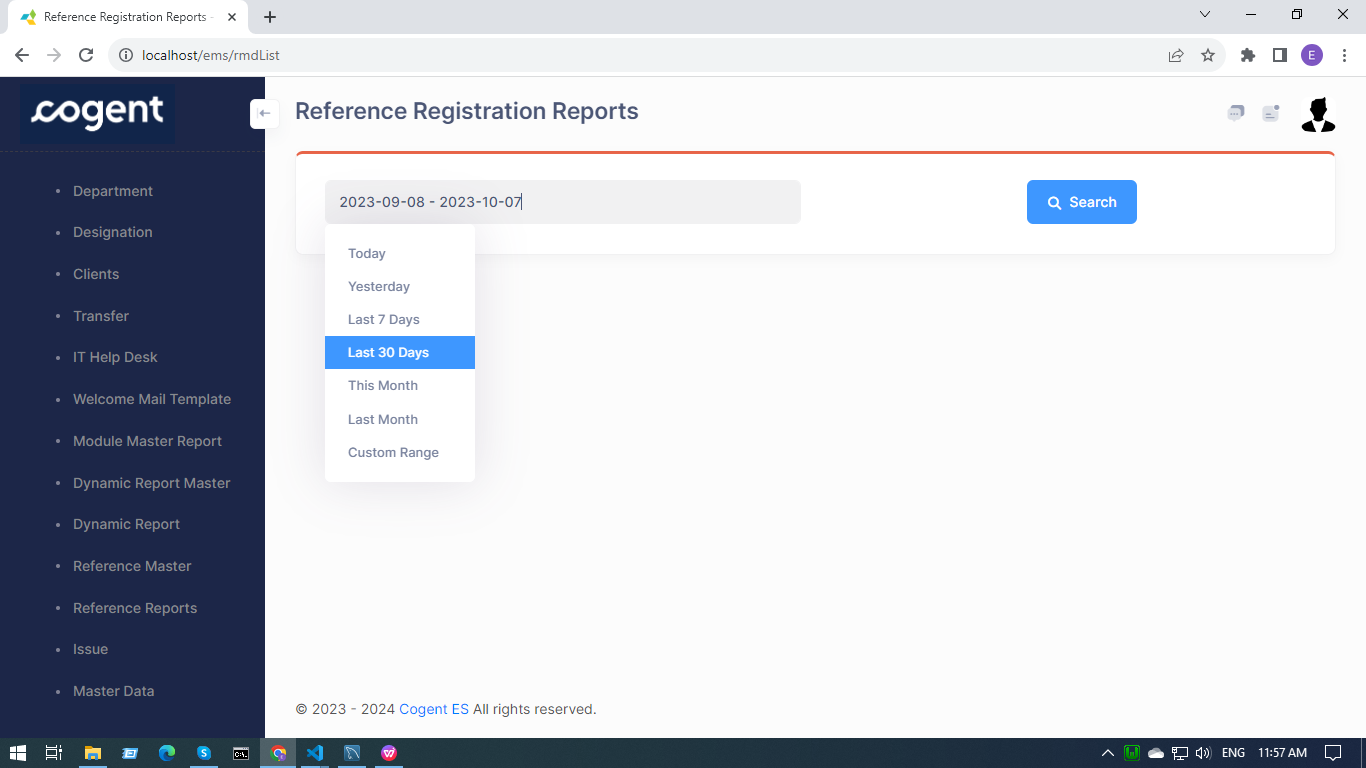
* Data fetched from the database is displayed in a data-table.
* Users can search for a particular entry.
* Users can export the data in Excel or CSV format.
* Users can delete a particular entry from the database.
* There are two main functions : Add and Edit Reference Master Details.
* Users can add reference details by clicking on the add button.



* Users can edit reference details by clicking on the edit button.
* Click the submit button to save the data in the database.



**Reference Reports :-**



Routes :-

use App\Http\Controllers\Rmd\ReferenceMaster;

/\*For Reference Registration Report Routes\*/

Route::any('rmdList', [ReferenceMaster::class, 'list'])->name('rmdList');

Controller :-

namespace App\Http\Controllers\Rmd;

use App\Http\Controllers\Controller;

use Illuminate\Support\Facades\Session;

use App\Models\Rmd\RefMaster;

use Illuminate\Http\Request;

use Exception;

use Illuminate\Support\Facades\DB;

Blade File :-

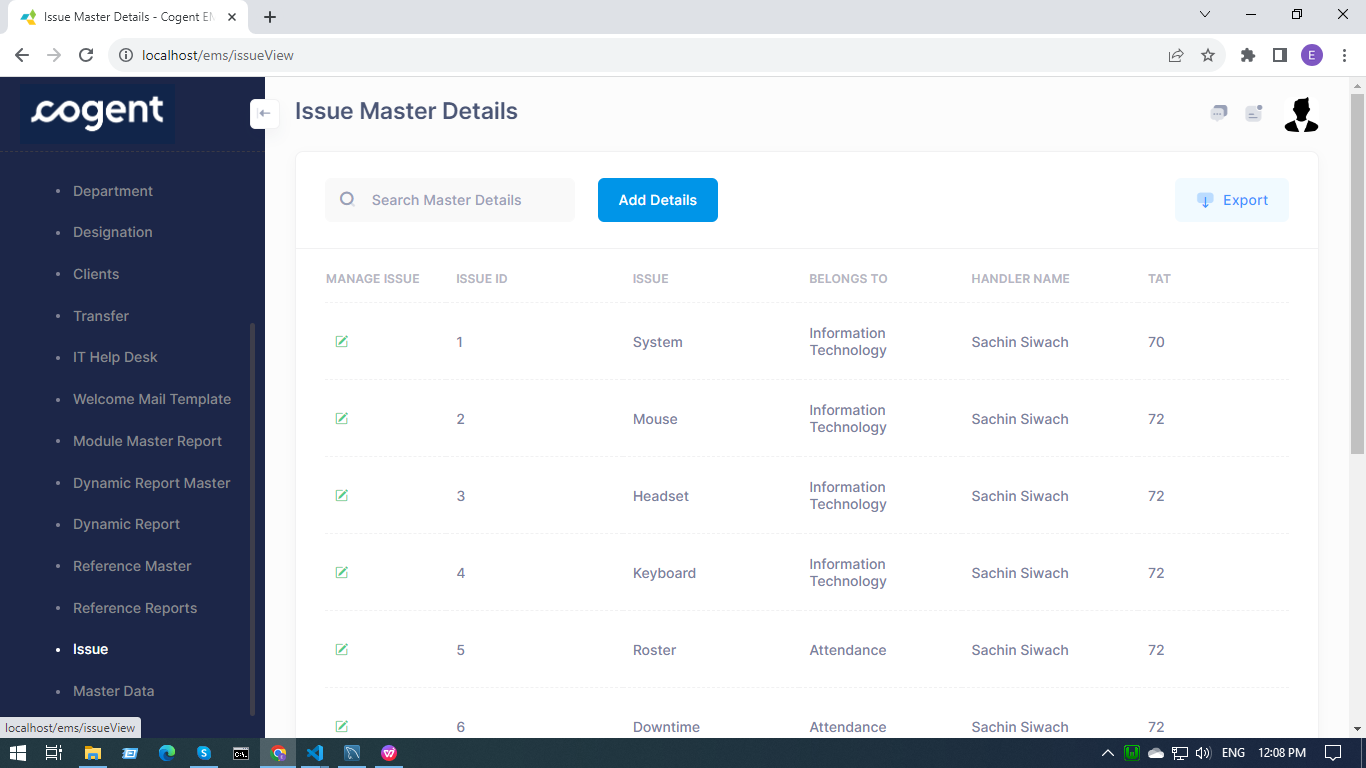
 return view('rmd/rmdList', compact('results'));

Function :-

* Data fetched from the database is displayed in a data-table.
* Users can search for a particular entry.
* Users can export the data in Excel or CSV format.
* The total entries in the database are < 45000, so the page loading time was exceeding the max\_execution\_time of Laravel I.e. 120 (2 min) so we are fetching only limited data-entries to avoid page crashing.

->take(10000)

**Issue :-**



Routes :-

use App\Http\Controllers\Issue\IssueMaster;

/\*For issue Master Details Routes\*/

Route::get('issueView', [IssueMaster::class, 'view'])->name('issueView');

Route::any('issueUpdate', [IssueMaster::class, 'update'])->name('issueUpdate');

Route::post('issueSave', [IssueMaster::class, 'save'])->name('issueSave');

Route::post('issueAdd', [IssueMaster::class, 'add'])->name('issueAdd');

Controller :-

namespace App\Http\Controllers\Issue;

use App\Http\Controllers\Controller;

use Illuminate\Http\Request;

use App\Models\Issue\Issue;

use Illuminate\Support\Facades\DB;

use Illuminate\Support\Facades\Session;

use Exception;

Model :-

namespace App\Models\Issue;

use Illuminate\Database\Eloquent\Factories\HasFactory;

use Illuminate\Database\Eloquent\Model;

class Issue extends Model

{

    use HasFactory;

    protected $table = "issue\_master";

    protected $fillable = ['id', 'queary', 'bt', 'handler', 'tat', 'modifiedby', 'modifiedon'];

    public $timestamps = false;

}

Blade File :-

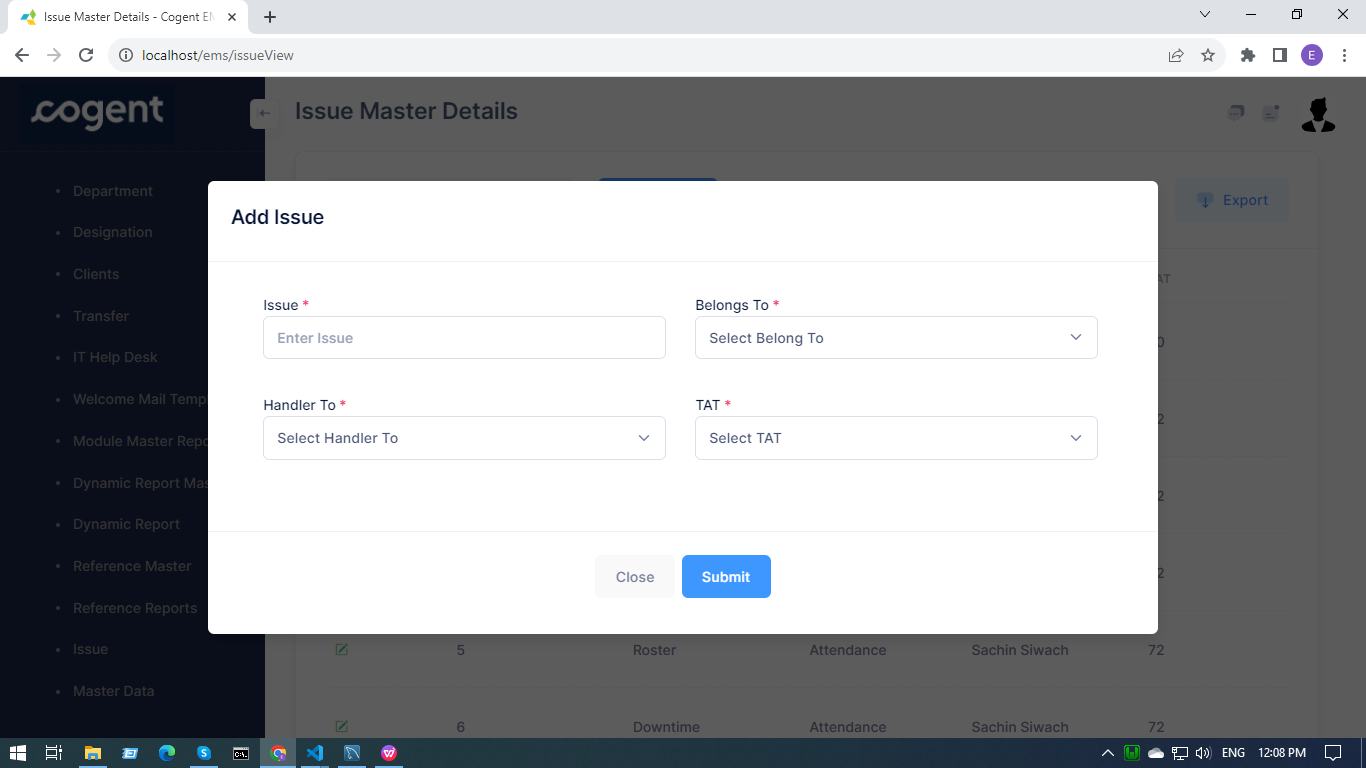
return view('issue/issueView', compact('results'));

Function :-

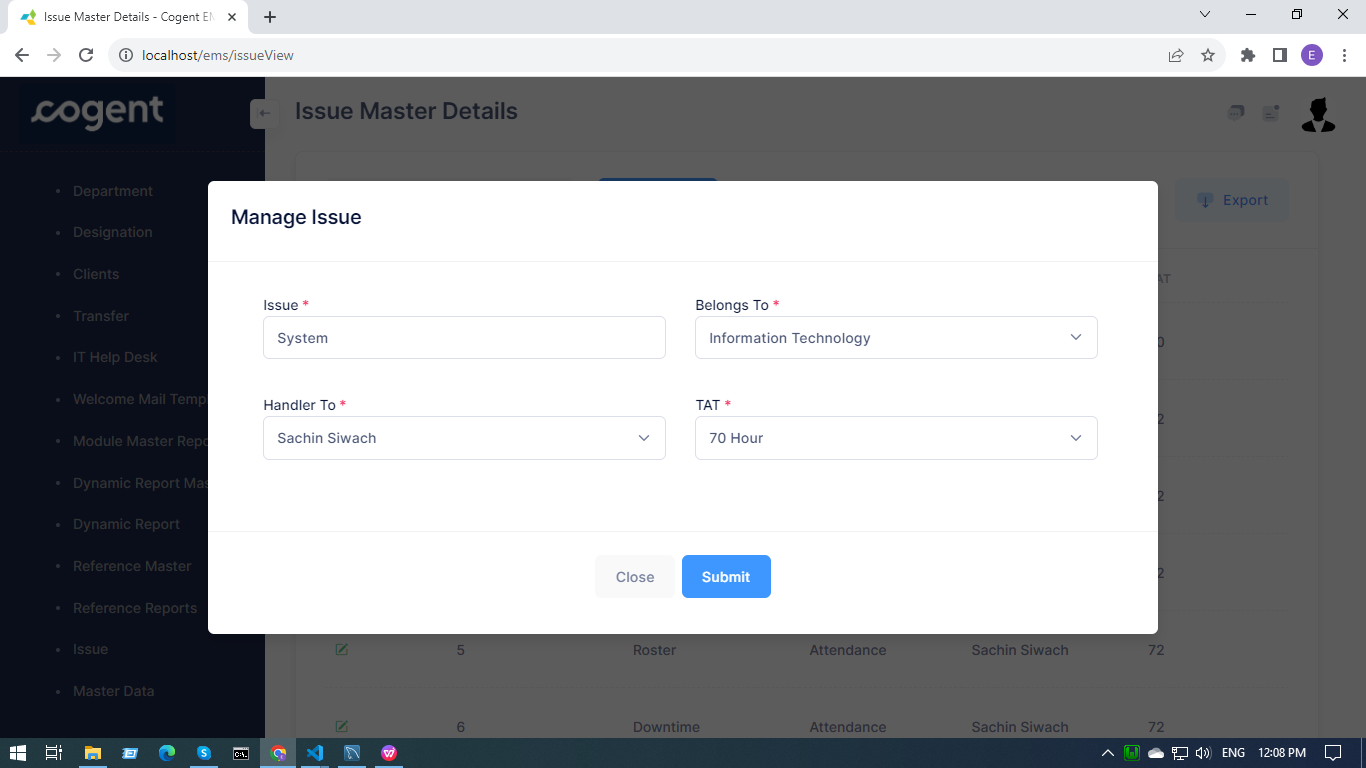
* Data fetched from the database is displayed in a data-table.
* Users can search for a particular entry.
* Users can export the data in Excel or CSV format.
* Users can delete a particular entry from the database.
* There are two main functions :

Add and Edit Issue Master Details.

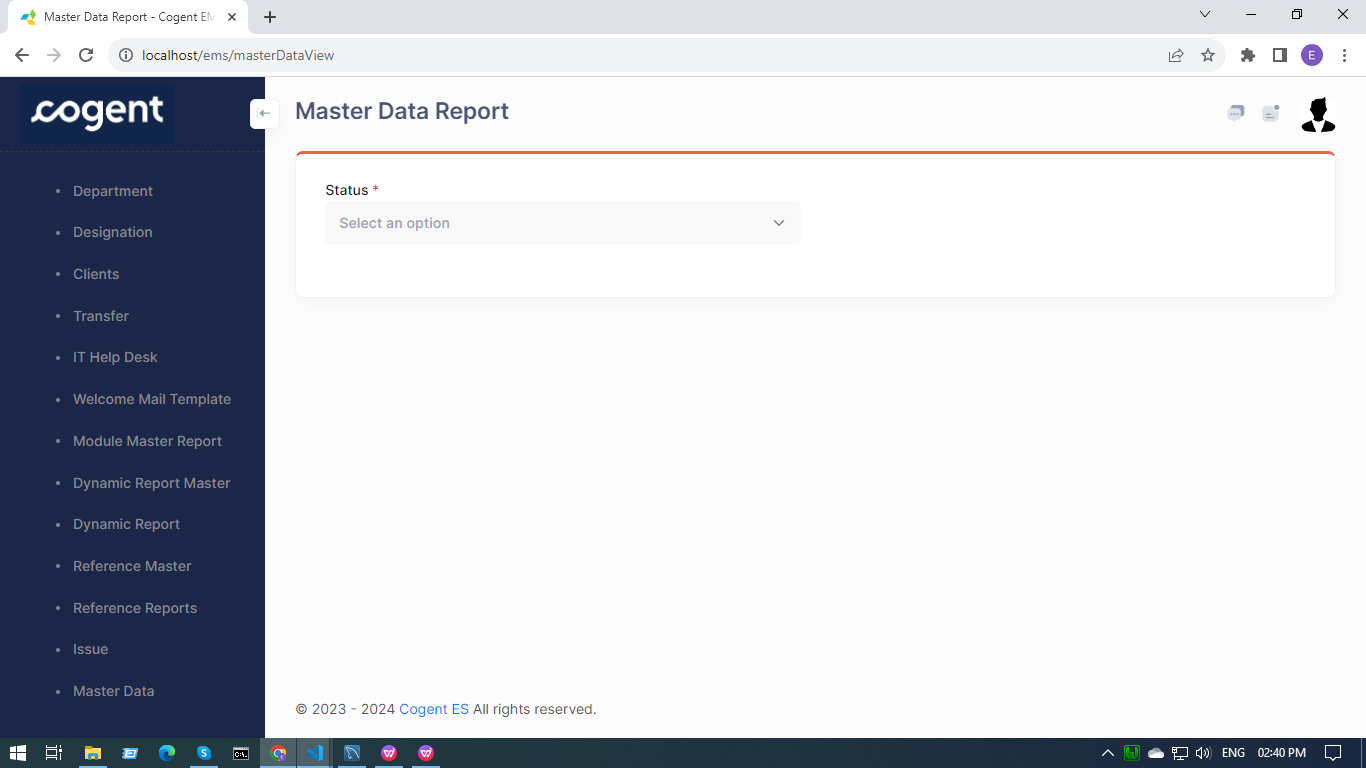
* Users can add issue details by clicking on the add button.
* Click on the save button to save the data into the database.



* Users can edit issue details by clicking on the edit button.



**Master Data :-**



Routes :-

use App\Http\Controllers\MasterData\DataController;

/\*For Master Data Report Routes\*/

Route::get('masterDataView', [DataController::class, 'view'])->name('masterDataView');

Route::any('getActiveData', [DataController::class, 'activeData'])->name('getActiveData');

Route::any('getInactiveData', [DataController::class, 'inactiveData'])->name('getInactiveData');

Controller :-

namespace App\Http\Controllers\MasterData;

use App\Http\Controllers\Controller;

use Illuminate\Http\Request;

use Illuminate\Support\Facades\DB;

use Exception;

Blade File :-

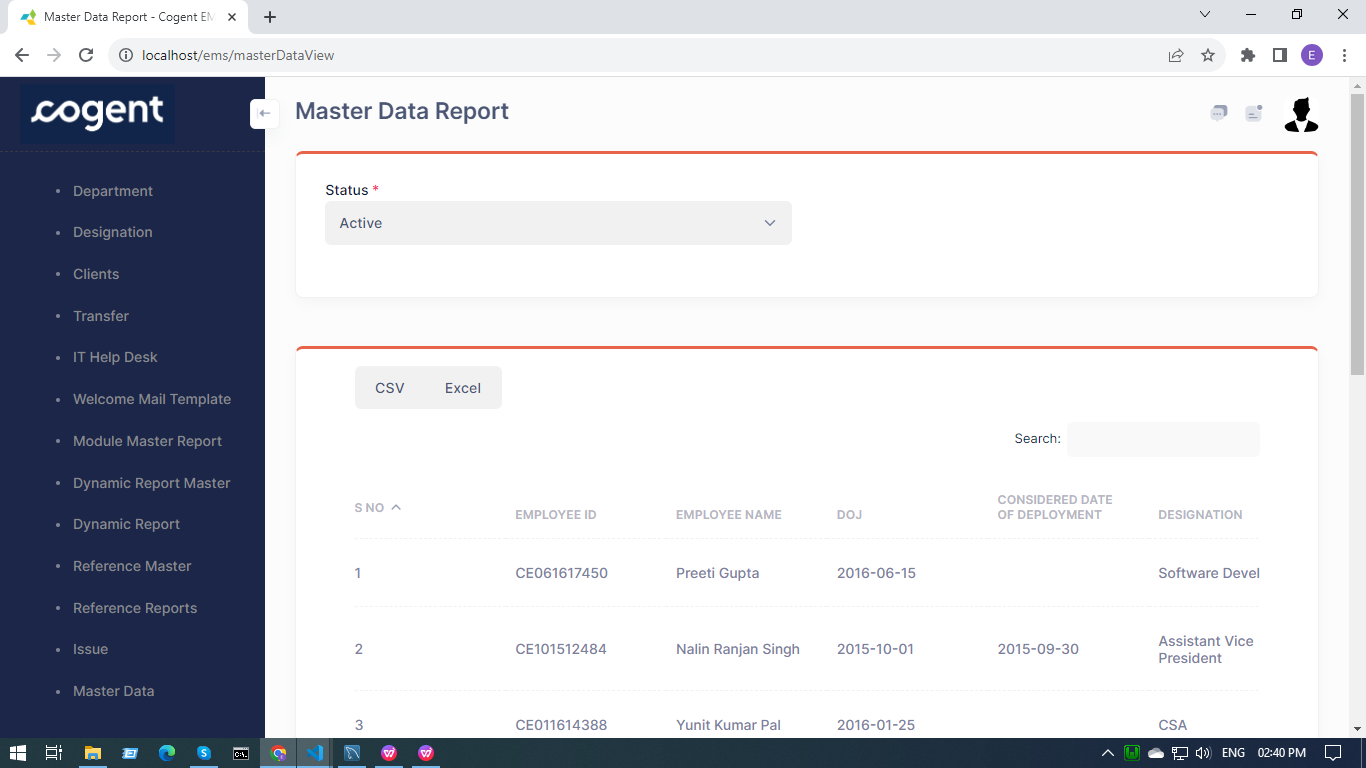
 return view('masterData/masterDataView');

Ajax :-

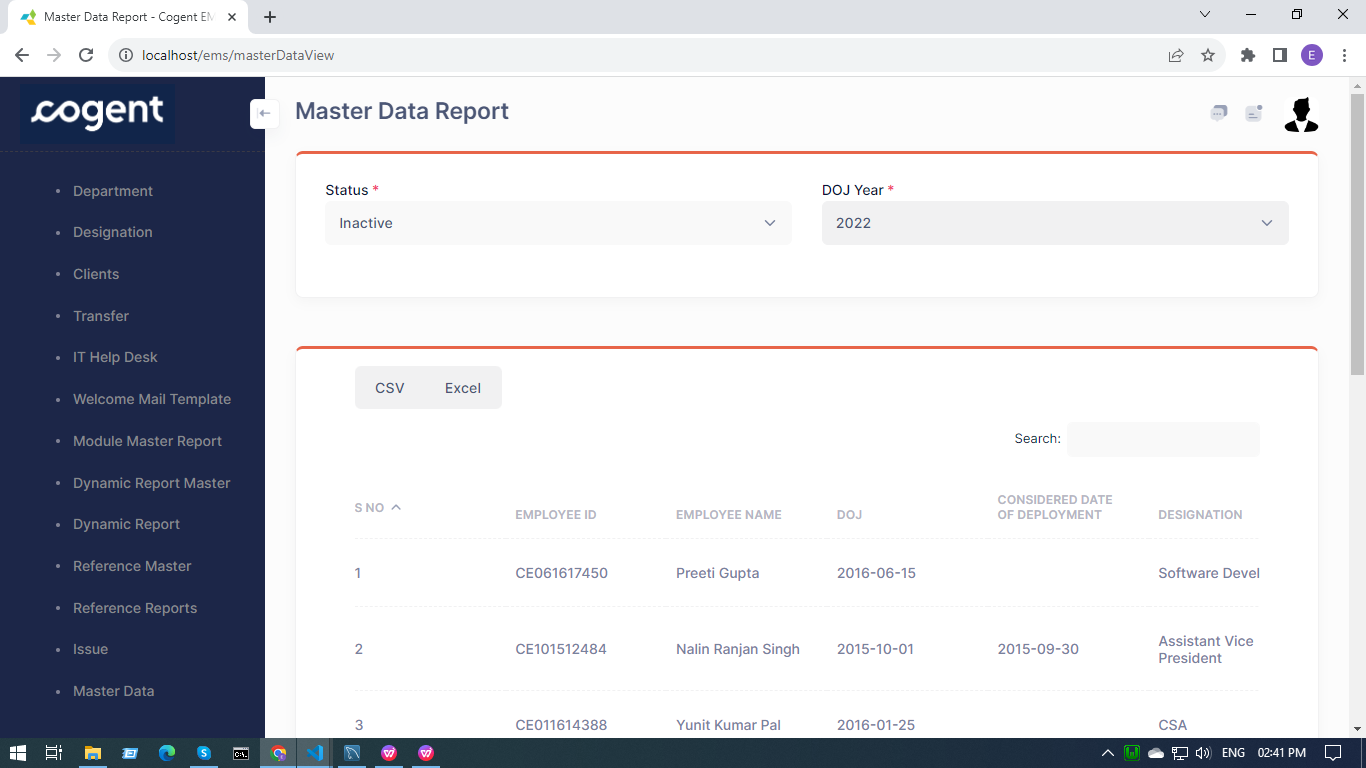
 return response()->json(['table' =>  view('ajax.masterdata.table', compact('result'))->render(), 'success' => true]);

Function :-

* Users on this page have to select the Status first for the data to be fetched from the database.
* Status can be either Active or Inactive for the employee.
* If the status is active :



* If the status is inactive then the user will select the year for which the data should be fetched from the database.



* Data fetched from the database is displayed in a data-table.
* Users can search for a particular entry.
* Users can export the data in Excel or CSV format.