TableBuilder/FormBuilder Dev Documentation

Version 1.0.1

Author Puneet Mehra

Reference Link - https://www.ag-grid.com/archive/21.2.0/best-angular-2-data-grid/

File structure

TableBuilderHelper - registered as a helper with the alias in the config/app.php It can be called from anywhere from the blade files or controller like as below. We will pass the module name from the route. Table builder helper is used to create fully dynamic table with the extended features.

File path

'FormBuilderHelper' => App\Helper\FormBuilderHelper::class,

'TableBuilderHelper' => App\Helper\TableBuilderHelper::class,

Route - Here module will be dynamic

Frontend URL - http://localhost:8000/table-builder?module=Employee

Route::get('/table-builder', [FormBuilderController::class, 'getDataForTable'])->name('getDataForTable');

Controller - Here passing two types of condition one is with Ajax and one is onload.

 public function getDataForTable(Request $request){

        if(isset($\_GET['module'])){

            $module = $\_GET['module'];

            $params = array();

            if ($request->ajax()) {

                if(isset($\_GET['type'])){

                    $params['pagination']= true;

                }

                TableBuilderHelper::tableBuilder($module,$params);

                return;

            }

        }

        return view('formbuilder.table',["module"=>$module]);

    }

When render first time below line of code will work

return view('formbuilder.table',["module"=>$module]);

View - resources\views\formbuilder\table.blade.php

It will include below line of code only to generate table for any module

 {{ TableBuilderHelper::tableBuilder($module) }}

Helper - app\Helper\TableBuilderHelper.php

Function - public static function tableBuilder($model,$params=null)

It is being used for all business logic to render table including search

Another important function. This function is being used to get available coloms and the colom order that need to be display in any table.

$page\_settings = self::getPageSettingsForUser(1,$model);

The settings are being save in page\_settings table

Below are the some predefined seeders

Colom - settings\_array contains colom\_order and colom\_show and page\_size for the particular user. Every user can have his own settings

Colom - permission\_id will be refer to the specific model

 DB::table('page\_settings')->insert([

            'user\_id'=>1,

            'permission\_id'=>2,

            'settings\_array'=>'[{"colom\_order":"2,1,3"},{"show\_colom":"1,2,3"},{"page\_size":"20"}]',

        ]);

Default settings - All model have some default settings.

Here we have an additional settings which will be coloms\_default For e.g we don’t want to show colom like created\_at,updated\_at or any primary key then we will not add in the coloms\_default.

For default settings user\_id always be 0

DB::table('page\_settings')->insert([

            'user\_id'=>0,

            'permission\_id'=>2,

            'settings\_array'=>'[{"colom\_order":"1,2,3"},{"show\_colom":"1,2,3"},{"page\_size":"10"},{"coloms\_default":"1,2,3,4"}]',

        ]);

Other funcion for search

getSyntaxByOperator

It will get operator by selected value from dropdown

 if($operator == "contains"){

            $syntax = 'like';

            $value = '%'.$value.'%';

        }

        if($operator == "not\_contains"){

            $syntax = 'Not like';

            $value = '%'.$value.'%';

        }

        if($operator == "equals"){

            $syntax = '=';

            $value = $value;

        }

        if($operator == "not\_equals"){

            $syntax = '<>';

        }

        if($operator == "ends\_with"){

            $syntax = 'Like';

            $value = '%'.$value;

        }

        if($operator == "starts\_with"){

            $syntax = 'Like';

            $value = $value.'%';

        }

getDefaultColoms - function is being used to get the default coloms of the module

public static function getDefaultColoms($permission\_id=null){

Views File

resources\views\layouts\tablebuilder.blade.php - It contains the table structure include the search functionality includes jquery in this file for Advance serach as well

resources\views\layouts\tableBody.blade.php - It is separate blade file included in tablebuilder because on every serach this file data will be refreshsed

resources\views\layouts\pagination.blade.php - It contains the pagination only which refresh on every search request

resources\views\layouts\pdfview.blade.php - It is being used for the Export to PDF functionality.

resources\views\Modal\TableSettingsModal.blade.php - It is the modal is being used for the colom configuration for the show /hide and colom order by

Pagination functionality - Currently Ajax pagination we are using with some customization over the top of laravel pagaintion

Colom Sorting - Currently we are using laravel sorting with some customization over the top of laravel sorting

Laravel query function for the advance filter as below

$data = $model::where(function($q) use ($whereArray) {

                        $i=0;

                        foreach($whereArray as $row) {

                            $q->Where($row['colname'], $row['operator'],$row['value']);

                            if(isset($row['advance\_op'])){

                                if($row['advance\_op']=="and"){

                                    $q->Where($row['colname'], $row['operatortwo'],$row['valuetwo']);

                                } else {

                                    $q->orWhere($row['colname'], $row['operatortwo'],$row['valuetwo']);

                                }

                            }

                            $i++;

                        }

                        // other conditions this

                    })->orderBy($column\_name, $order\_type)->sortable()->paginate($page\_size,['\*'],'custom-page',$page);

Another important points for security prospective - we are using colom index only to pass the search parameters from the view to controller using ajax instead of direcct colom name

Below is the naming convention for the search input. For e.g if we have colom firstname with the index one. Then we will pass key as 1 for the serach firstname instead of the colom name

name = "search[coloms][opvalue]{{$key}}"