<?php

namespace App\Http\Controllers;

use App\Group;

use App\Idea;

use App\Proposal;

use App\Supervisor;

use Auth;

use DB;

use Illuminate\Http\Request;

use \Atomescrochus\StringSimilarities\Compare;

class ProposalsController extends Controller

{

/\*\*

\* Display a listing of the resource.

\*

\* @return \Illuminate\Http\Response

\*/

public function index()

{

//similar\_text('bafoobar', 'barfoo', $perc);

//Proposals for GPC's view using eloquent

$proposals = Proposal::all();

$comparison = new \Atomescrochus\StringSimilarities\Compare();

//Condition to allow only GPC to perform this kind of processes.

if (Auth::user()->user\_role == 2) {

//Two for loops,the dirst is to count through the proposal. The second is to compare proposal of first loop with other proposals.

for ($i = 0; $i < count($proposals); $i++) {

$max = 0;

for ($x = 0; $x < count($proposals); $x++) {

//This condition prevents a proposal to be compared with it-self

if ($x != $i) {

$jaroWinkler = $comparison->smg($proposals[$i]->describtion, $proposals[$x]->describtion);

// $proposals[$i]->describtion." / ".$proposals[$x]->describtion;

if ($max < $jaroWinkler) {

$max = $jaroWinkler;

$p\_id = $proposals[$x]->id;

}

} else { $p\_id = "None";}

// return $jaroWinkler;

}

//Multiplying number by 100 to get the percentage

$max = $max \* 100;

//Casting the number and specifying the format.

$max = number\_format($max, 2, '.', '');

$proposals[$i]->setAttribute('Similarity', $max);

$proposals[$i]->setAttribute('P\_ID', $p\_id);

}

return view('tables', compact('proposals'));

} else

//Proposals for students' and supervisors' view using Raw DB

{

$proposalsSTU = DB::select(DB::raw("SELECT proposals.id, proposals.title, proposals.status ,proposals.supervisor\_id from proposals"));

}

foreach ($proposals as $proposal) {

// $Count = array(["arrayname" => "bla bla bla"]);

$Count = Idea::where('proposal\_id', $proposal->id)->get()->count();

// $proposal= Arr::prepend($proposal, $Count, 'Count');

$proposal->setAttribute('Times', $Count);

}

// $proposals->push($Count);

// return $proposals;

return view('tables', compact('proposalsSTU', 'proposals'));

}

/\*\*

\* Show the form for creating a new resource.

\*

\* @return \Illuminate\Http\Response

\*/

public function create()

{

//

return view('ProposeIdea');

}

/\*\*

\* Store a newly created resource in storage.

\*

\* @param \Illuminate\Http\Request $request

\* @return \Illuminate\Http\Response

\*/

public function store(Request $request)

{

//

$user = Auth::user();

if ($user->user\_role == 3) {

//Validating the fields of the form that are required.

$request->validate([

'Title' => 'required|min:10',

'Describtion' => 'required|min:10',

'requiredResources' => 'required|min:10',

]);

$proposals = new Proposal();

$proposals->Title = $request->input('Title');

$proposals->Describtion = $request->input('Describtion');

$proposals->requiredResources = $request->input('requiredResources');

$proposals->supervisor\_id = Auth::user()->id;

$proposals->status = false;

// $proposals->user\_id = $user->id;

$proposals->save();

return redirect('proposals')->with('success', 'Proposal created successfully');

}

//In here GPC Will approve or deny

elseif ($user->user\_role == 2) {

if ($request->Button == 'Approve') {

$id = $request->proposal\_id;

$proposal = Proposal::find($id);

$proposal->status = true;

$proposal->save();

return view('proposalshow', compact('proposal'));

} elseif ($request->Button == 'Reject') {

$proposal = Proposal::find($request->proposal\_id);

$proposal->status = false;

$proposal->save();

return view('proposalshow', compact('proposal'));

}

} elseif ($user->user\_role == 5) {

$GID = DB::table('students')->where('id', Auth::user()->id)->value('group\_id');

//Retrieving the current user's group\_id

$group = Group::find($GID);

//Only group leader can perform the proposal selection process.

if ($group->Leader == Auth::user()->id) {

$Idea = new Idea();

$Idea->Proposal\_id = $request->proposal\_id;

$Idea->Group\_id = $GID;

$Idea->Final = 0;

$Idea->save();

return redirect('proposals')->with('SuccessMessage', 'Proposal chosen successfully');

} else {

return redirect('proposals')->with('ErrorMessage', "You are not your group's leader ! only group leader can choose a proposal");

}

}

}

/\*\*

\* Display the specified resource.

\*

\* @param int $id

\* @return \Illuminate\Http\Response

\*/

public function show($id)

{

//

$proposal = Proposal::find($id);

return view('proposalshow', compact('proposal'));

}

/\*\*

\* Show the form for editing the specified resource.

\*

\* @param int $id

\* @return \Illuminate\Http\Response

\*/

public function edit($id)

{

//

$proposal = Proposal::find($id);

$userID = Auth::user()->id;

// return $id;

if ($proposal->supervisor->id !== $userID) {

return Redirect('proposals')->withErrors(['Error !', 'This is not your proposal !']);

} else {

return view('proposaledit', compact('proposal'));

}

}

/\*\*

\* Update the specified resource in storage.

\*

\* @param \Illuminate\Http\Request $request

\* @param int $id

\* @return \Illuminate\Http\Response

\*/

public function update(Request $request, $id)

{

//

$request->validate([

'Title' => 'required|min:10',

'Describtion' => 'required|min:10',

'requiredResources' => 'required|min:10',

]);

$proposal = Proposal::find($id);

$proposal->Title = $request->input('Title');

$proposal->Describtion = $request->input('Describtion');

$proposal->requiredResources = $request->input('requiredResources');

$proposal->save();

return redirect('/proposals/' . $proposal->id);

}

/\*\*

\* Remove the specified resource from storage.

\*

\* @param int $id

\* @return \Illuminate\Http\Response

\*/

public function destroy($id)

{

//

//Finding the proposalby id using Eloquent.

$proposal = Proposal::find($id);

$userID = Auth::user()->id;

//Only supervisor can perform the deletion process.

if ($proposal->supervisor->id !== $userID) {

return Redirect('proposals')->withErrors(['Error !', 'This is not your proposal !']);

} else {

$proposal->delete();

return redirect('proposals')->with('SuccessMessage', 'Proposal deleted successfully');

}

}

public function ViewChosenProposal()

{

//Getting the details of the approved proposal and showing the details of the group/groups that chose them

$CurrUser = Auth::user()->id;

$GroupsThatChoseProposals = DB::select(DB::raw("SELECT DISTINCT ideas.group\_id FROM ideas, proposals, supervisors

WHERE proposals.id = (select id from proposals where supervisor\_id = $CurrUser )"));

//\*\*here we r retriving the group info who select the supervisor proposal \*/

$GroupNumber = count($GroupsThatChoseProposals);

foreach ($GroupsThatChoseProposals as $GroupsThatChoseProposal) {

$group1 = $GroupsThatChoseProposals[0]->group\_id;

$group1Members = DB::select(DB::raw("SELECT students.id, students.first\_name, students.last\_name, departments.name FROM

students, departments WHERE group\_id = $group1 AND departments.id = students.department\_id "));

//\*\* if 2 gruop select the proposals \*/

if ($GroupNumber == 2) {

//Taking the id of the second group from the query (if there is any)

$group2 = $GroupsThatChoseProposals[1]->group\_id;

$group2Members = DB::select(DB::raw("SELECT students.id, students.first\_name, students.last\_name, departments.name FROM

students, departments WHERE group\_id = $group2 AND departments.id = students.department\_id "));

}

}

//I don't know what the heck is hapening in here !

if ($GroupNumber == 2) {

$id = DB::table('ideas')->where('group\_id', $group1)->orWhere('group\_id', $group2)->value('id');

$Check = DB::table('ideas')->where('id', $id)->value('Final');

} elseif ($GroupNumber == 1) {

$id = DB::table('ideas')->where('group\_id', $group1)->value('id');

$Check = DB::table('ideas')->where('id', $id)->value('Final');

}

if ($GroupNumber == 2) {

return view('tables-details', compact('group2Members', 'group1Members', 'GroupNumber', 'group1', 'group2', 'Check'));

// return $group1Members;

} elseif ($GroupNumber == 1) {

return view('tables-details', compact('group1Members', 'GroupNumber', 'group1', 'Check'));

} else {

return redirect('/proposals')->with('WarningMessage', 'No groups have chosen this proposal yet !');

}

// return $GroupsThatChoseProposals[0]->group\_id;

}

public function FinalApprove(Request $request)

{

if (Auth::user()->user\_role == 3) {

//Getting the id of the finalized proposal

$id = DB::table('ideas')->where('group\_id', $request->Group\_id)->value('id');

//Check if the proposal is already finalized

$Check = DB::table('ideas')->where('id', $id)->value('Final');

if ($Check == 1) {

return redirect('/')->with('ErrorMessage', 'A group has already been finalized to this proposal');

} else {

$idea = Idea::find($id);

$idea->Final = true;

$idea->save();

$supervisor = Supervisor::find(Auth::user()->id);

$supervisor->group\_id = $request->Group\_id;

}

}

}

}