```php
<?php

namespace App\Controllers;

use TCPDF;

use Core\{Controller, H, Session, Router, DB};
use App\Models\{Projects, Users, Clients, Clusters, ProjectBudgetlineRefs,
Budgetlines, Companies, Countries, Currencies, EmployeePbrs, PayrollTemplate,
Employees, GlAccounts, PayrollDetails, PayrollItems, Payrolls, ProjectEmployees,
TaxBands, VoucherGlRefs, Vouchers};

class PayrollsController extends Controller
{
    public function onConstruct()
    {
        $this->view->company = Companies::findById(1);
        $this->view->setLayout('admin');
        $this->currentUser = Users::currentUser();
    }

    public function processprojectpayroll2Action($payroll_id)
    {

        $payroll = Payrolls::findById((int)$payroll_id);
        $project = Projects::findByIdAndActiveStatus((int)$payroll->project_id);
        $user = Users::currentUser();
        if (!$payroll) {
            Session::addMsg('danger', 'You must first initiate thr project
payroll, fill up details below to continue');
            Router::redirect('payrolls/initiateprojectpayroll');
        }

        if (!$project) {
            Session::addMsg('danger', 'The project for this specific payroll is
inactive, activate the project before you proceed');
            Router::redirect('payrolls/index');
        }

        if ($this->request->isPost()) {
            $this->request->csrfCheck();
            $empListArray = explode('^^', $this->request->get('empList'));
            $payrollListArray = explode('^^', $this->request-
>get('payroll_List'));
            $prolItemsTotals = array_fill(0, count($payrollListArray), 0.00);
            $mappings = [];
            for ($a = 0; $a < count($empListArray); $a++) {
                for ($c = 0; $c < count($payrollListArray); $c++) {

                    if (!$this->request->get('amount_' . $empListArray[$a] . "_"
. $payrollListArray[$c])) {
                        continue;
                    }
                    $payroll_details_id = $this->request->get('id_' .
$empListArray[$a] . "_" . $payrollListArray[$c]);
                    $payroll_details = ($payroll_details_id == "") ? new
PayrollDetails() : PayrollDetails::findById($payroll_details_id);
                    $payroll_details->payroll_id = $payroll->id;
                    $payroll_details->employee_id = $empListArray[$a];
                    $payroll_details->project_id = $project->id;
                    $payroll_details->project_budgetline_ref_id  = $this-
>request->get('budgetLine_' . $empListArray[$a]);
                    $payroll_details->rate = $this->request-
>get('budgetLineAlloc_' . $empListArray[$a]);
```

```
                    $payroll_details->monthly_salary = $this->request-
>get('monthly_salary_' . $empListArray[$a]);
                    $payroll_details->tax_auth_id = $this->request-
>get('taxAuth_' . $empListArray[$a]);
                    $payroll_details->item_id = $payrollListArray[$c];
                    $payroll_details->amount = $this->request->get('amount_' .
$empListArray[$a] . "_" . $payrollListArray[$c]);
                    $prolItemsTotals[$c] += $this->request->get('amount_' .
$empListArray[$a] . "_" . $payrollListArray[$c]);
                    $payrollItem =
PayrollItems::findById($payrollListArray[$c]);
                    if (!$payrollItem->debit_account_id && !$payrollItem-
>credit_account_id) {
                        continue;
                    }

                    if ($prolItemsTotals[$c] > 0) {
                        $mappings[] = [
                            'voucher_id' => '',
                            'voucher_ref_no' => '',
                            'payroll_id' => $payroll->id,
                            'project_name' => $project->name,
                            'payroll_ref_no' => $payroll->ref_no,
                            'payroll_item_id' => $payrollListArray[$c],
                            'payroll_item_name' => $payrollItem->code . '::' .
$payrollItem->name,
                            'debit_account_id' => $payrollItem-
>debit_account_id,
                            'credit_account_id' => $payrollItem-
>credit_account_id,
                            'totalAmount' => $prolItemsTotals[$c]
                        ];
                    }

                    $payroll_details->save();
                }
            }
            $payroll->is_posted = 1;
            $payroll->prl_voucher_data = json_encode($mappings);
            $payroll->save();
            $counter2 = 0;
            foreach ($mappings as $mapping) {
                // Vouchers Creation Section
                $voucher = Vouchers::findOrCreate();
                $voucher->document_id = $payroll->id;
                $voucher->ref_no = $voucher->ref_no;
                $voucher->trx_type = 4;
                $voucher->branch_id = 1;
                $voucher->currency_id = $payroll->currency_id;
                $voucher->country_id = $payroll->country_id;
                $voucher->doc_type = 4;
                $voucher->user_id = $user->id;
                $voucher->valid_date = date("Y-m-d");
                $voucher->description = "Payroll Voucher for " .
$mapping['payroll_item_name'];
                $voucher->save();
                // First row-Add debit row if debit account exists
                if ($mapping["debit_account_id"]) {
                    $voucher_gl_item = new VoucherGlRefs();
                    $voucher_gl_item->voucher_id = $voucher->id;
                    $voucher_gl_item->gl_account_id =
$mapping["debit_account_id"];
                    $voucher_gl_item->entry_type = "D";
                    $voucher_gl_item->currency_id = $voucher->currency_id;
                    $voucher_gl_item->amount_debit = $mapping["totalAmount"];
                    $voucher_gl_item->amount_credit = 0.00;
                    $voucher_gl_item->valid_date = $voucher->valid_date;
                    $voucher_gl_item->has_budgetlines = $voucher-
>has_budgetlines;
                    $counter2++;
                    $voucher_gl_item->save();
                }

                // Second row-Add credit row if credit account exists
                if ($mapping["credit_account_id"]) {
                    $voucher_gl_item = new VoucherGlRefs();
                    $voucher_gl_item->voucher_id = $voucher->id;
                    $voucher_gl_item->gl_account_id =
$mapping["credit_account_id"];
                    $voucher_gl_item->entry_type = "C";
                    $voucher_gl_item->currency_id = $voucher->currency_id;
                    $voucher_gl_item->amount_debit = 0.00;
                    $voucher_gl_item->amount_credit = $mapping["totalAmount"];
                    $voucher_gl_item->valid_date = $voucher->valid_date;
                    $voucher_gl_item->has_budgetlines = $voucher-
>has_budgetlines;
                    $counter2++;
                    $voucher_gl_item->save();
                }
            }
            // Redirects
            Session::addMsg('success', 'Payroll Processed and Saved
Successfully!');
            Router::redirect('payrolls/index');
        }
        $this->view->user = $user;
        $this->view->project_id = $project->id;
        $this->view->project_name = $project->code . '::' . $project->name;
        $this->view->payroll = $payroll;
        $this->view->payroll_id = $payroll->id;
        $this->view->payroll_ref_no = $payroll->ref_no;
        $this->view->gl_account_options = GlAccounts::getOptionsForForm();
        $this->view->entry_type_options = ['D' => 'Debit', 'C' => 'Credit'];
        $this->view->setLayout('admin');
        $this->view->render('payrolls/processprojectpayroll2');
    }
}
```