**E:\workspace\payroll\_rails\config\application.rb**

require File**.**expand\_path**(**'../boot'**,** **\_\_FILE\_\_)**

require 'rails/all'

**if** **defined?(**Bundler**)**

# If you precompile assets before deploying to production, use this line

Bundler**.**require**(\***Rails**.**groups**(**:assets **=>** %w(development test)**))**

# If you want your assets lazily compiled in production, use this line

# Bundler.require(:default, :assets, Rails.env)

**end**

**module** **Payroll**

**class** **Application** **<** Rails**::**Application

# Settings in config/environments/\* take precedence over those specified here.

# Application configuration should go into files in config/initializers

# -- all .rb files in that directory are automatically loaded.

# Custom directories with classes and modules you want to be autoloadable.

# config.autoload\_paths += %W(#{config.root}/extras)

# Only load the plugins named here, in the order given (default is alphabetical).

# :all can be used as a placeholder for all plugins not explicitly named.

# config.plugins = [ :exception\_notification, :ssl\_requirement, :all ]

# Activate observers that should always be running.

# config.active\_record.observers = :cacher, :garbage\_collector, :forum\_observer

# Set Time.zone default to the specified zone and make Active Record auto-convert to

# this zone.

# Run "rake -D time" for a list of tasks for finding time zone names. Default is UTC.

# config.time\_zone = 'Central Time (US & Canada)'

config**.**time\_zone **=** 'Kuala Lumpur'

# The default locale is :en and all translations from config/locales/\*.rb,yml are auto

# loaded.

# config.i18n.load\_path += Dir[Rails.root.join('my', 'locales', '\*.{rb,yml}').to\_s]

# config.i18n.default\_locale = :de

# Configure the default encoding used in templates for Ruby 1.9.

config**.**encoding **=** "utf-8"

# Configure sensitive parameters which will be filtered from the log file.

config**.**filter\_parameters **+=** **[**:password**]**

# Enable escaping HTML in JSON.

config**.**active\_support**.**escape\_html\_entities\_in\_json **=** **true**

# Use SQL instead of Active Record's schema dumper when creating the database.

# This is necessary if your schema can't be completely dumped by the schema dumper,

# like if you have constraints or database-specific column types

# config.active\_record.schema\_format = :sql

# Enforce whitelist mode for mass assignment.

# This will create an empty whitelist of attributes available for mass-assignment for

# all models in your app. As such, your models will need to explicitly whitelist or

# blacklist accessible parameters by using an attr\_accessible or attr\_protected

# declaration.

config**.**active\_record**.**whitelist\_attributes **=** **true**

# Enable the asset pipeline

config**.**assets**.**enabled **=** **true**

# Version of your assets, change this if you want to expire all your assets

config**.**assets**.**version **=** '1.0'

config**.**assets**.**precompile **+=** %w(loginui.css \_payslip.css \_payslip.js

dark-hive/jquery-ui-1.10.2.custom.min.css admin.css admin.js user.css user.js)

jquitheme **=** %w(blitzer dark-hive trontastic humanity)

jquicss **=** 'jquery-ui-1.10.2.custom.min.css'

jquitheme**.**each **do** **|**t**|**

config**.**assets**.**precompile **<<** "**#{**t**}**/**#{**jquicss**}**"

**end**

**end**

**end**

**E:\workspace\payroll\_rails\config\database.yml**

# MySQL. Versions 4.1 and 5.0 are recommended.

#

# Install the MYSQL driver

# gem install mysql2

#

# Ensure the MySQL gem is defined in your Gemfile

# gem 'mysql2'

#

# And be sure to use new-style password hashing:

# http://dev.mysql.com/doc/refman/5.0/en/old-client.html

**development**:

**adapter**: mysql2

**encoding**: utf8

**reconnect**: false

**database**: payroll\_development

**pool**: 5

**username**: root

**password**: root

**host**: 127.0.0.1

**port**: 3307

# Warning: The database defined as "test" will be erased and

# re-generated from your development database when you run "rake".

# Do not set this db to the same as development or production.

**test**:

**adapter**: mysql2

**encoding**: utf8

**reconnect**: false

**database**: payroll\_test

**pool**: 5

**username**: root

**password**: root

**host**: 127.0.0.1

**port**: 3307

**production**:

**adapter**: mysql2

**encoding**: utf8

**reconnect**: false

**database**: payroll\_development

**pool**: 5

**username**: root

**password**: root

**host**: 127.0.0.1

**port**: 3307

**E:\workspace\payroll\_rails\config\environment.rb**

# Load the rails application

require File**.**expand\_path**(**'../application'**,** **\_\_FILE\_\_)**

# Initialize the rails application

Payroll**::**Application**.**initialize!

Rails**.**logger **=** Logger**.**new**(STDOUT)**

**E:\workspace\payroll\_rails\config\routes.rb**

Payroll**::**Application**.**routes**.**draw **do**

root :to **=>** 'admin/admin#index'

match 'login' **=>** 'application#new'**,** :as **=>** :login

match 'auth' **=>** 'application#create'**,** :as **=>** :auth

match 'logout' **=>** 'application#destroy'**,** :as **=>** :logout

namespace :admin **do**

match 'index' **=>** 'admin#index'**,** :as **=>** :index**,** :via **=>** :get

scope 'user'**,** :as **=>** 'user' **do**

match '' **=>** 'user#index'**,** :via **=>** :get

match 'list' **=>** 'user#list'**,** :as **=>** :list**,** :via **=>** **[**:get**,** :post**]**

match 'new' **=>** 'user#new'**,** :as **=>** :new**,** :via **=>** :get

match 'create' **=>** 'user#create'**,** :as **=>** :create**,** :via **=>** :post

match 'edit(/:id)' **=>** 'user#edit'**,** :as **=>** :edit**,** :via **=>** :get

match 'update(/:id)' **=>** 'user#update'**,** :as **=>** :update**,** :via **=>** :post

match 'delete' **=>** 'user#destroy'**,** :as **=>** :delete**,** :via **=>** :post

**end**

scope 'employee'**,** :as **=>** 'employee' **do**

match '' **=>** 'employee#index'**,** :via **=>** :get

match 'list' **=>** 'employee#list'**,** :via **=>** **[**:get**,** :post**]**

match 'new' **=>** 'employee#new'**,** :as **=>** :new**,** :via **=>** :get

match 'create' **=>** 'employee#create'**,** :as **=>** :create**,** :via **=>** :post

match 'edit(/:id)' **=>** 'employee#edit'**,** :as **=>** :edit**,** :via **=>** :get

match 'update(/:id)' **=>** 'employee#update'**,** :as **=>** :update**,** :via **=>** :post

match 'delete' **=>** 'employee#destroy'**,** :as **=>** :delete**,** :via **=>** :post

**end**

scope 'designation'**,** :as **=>** 'designation' **do**

match '' **=>** 'designation#index'**,** :via **=>** :get

match 'list' **=>** 'designation#list'**,** :as **=>** :list**,** :via **=>** **[**:get**,** :post**]**

match 'new' **=>** 'designation#new'**,** :as **=>** :new**,** :via **=>** :get

match 'create' **=>** 'designation#create'**,** :as **=>** :create**,** :via **=>** :post

match 'edit(/:id)' **=>** 'designation#edit'**,** :as **=>** :edit**,** :via **=>** :get

match 'update(/:id)' **=>** 'designation#update'**,** :as **=>** :update**,** :via **=>** :post

match 'delete' **=>** 'designation#destroy'**,** :as **=>** :delete**,** :via **=>** :post

**end**

scope 'empstatus'**,** :as **=>** 'empstatus' **do**

match '' **=>** 'employment\_status#index'**,** :via **=>** :get

match 'list' **=>** 'employment\_status#list'**,** :as **=>** :list**,** :via **=>** **[**:get**,** :post**]**

match 'new' **=>** 'employment\_status#new'**,** :as **=>** :new**,** :via **=>** :get

match 'create' **=>** 'employment\_status#create'**,** :as **=>** :create**,** :via **=>** :post

match 'edit(/:id)' **=>** 'employment\_status#edit'**,** :as **=>** :edit**,** :via **=>** :get

match 'update(/:id)' **=>** 'employment\_status#update'**,** :as **=>** :update**,** :via **=>** :post

match 'delete' **=>** 'employment\_status#destroy'**,** :as **=>** :delete**,** :via **=>** :post

**end**

scope 'jobcat'**,** :as **=>** 'jobcat' **do**

match '' **=>** 'job\_category#index'**,** :via **=>** :get

match 'list' **=>** 'job\_category#list'**,** :as **=>** :list**,** :via **=>** **[**:get**,** :post**]**

match 'new' **=>** 'job\_category#new'**,** :as **=>** :new**,** :via **=>** :get

match 'create' **=>** 'job\_category#create'**,** :as **=>** :create**,** :via **=>** :post

match 'edit(/:id)' **=>** 'job\_category#edit'**,** :as **=>** :edit**,** :via **=>** :get

match 'update(/:id)' **=>** 'job\_category#update'**,** :as **=>** :update**,** :via **=>** :post

match 'delete' **=>** 'job\_category#destroy'**,** :as **=>** :delete**,** :via **=>** :post

**end**

scope 'dept'**,** :as **=>** 'dept' **do**

match '' **=>** 'department#index'**,** :via **=>** :get

match 'list' **=>** 'department#list'**,** :as **=>** :list**,** :via **=>** **[**:get**,** :post**]**

match 'new' **=>** 'department#new'**,** :as **=>** :new**,** :via **=>** :get

match 'create' **=>** 'department#create'**,** :as **=>** :create**,** :via **=>** :post

match 'edit(/:id)' **=>** 'department#edit'**,** :as **=>** :edit**,** :via **=>** :get

match 'update(/:id)' **=>** 'department#update'**,** :as **=>** :update**,** :via **=>** :post

match 'delete' **=>** 'department#destroy'**,** :as **=>** :delete**,** :via **=>** :post

**end**

scope 'payrate'**,** :as **=>** 'payrate' **do**

match '' **=>** 'pay\_rate#index'**,** :via **=>** :get

match 'list' **=>** 'pay\_rate#list'**,** :as **=>** :list**,** :via **=>** **[**:get**,** :post**]**

match 'new' **=>** 'pay\_rate#new'**,** :as **=>** :new**,** :via **=>** :get

match 'create' **=>** 'pay\_rate#create'**,** :as **=>** :create**,** :via **=>** :post

match 'edit(/:id)' **=>** 'pay\_rate#edit'**,** :as **=>** :edit**,** :via **=>** :get

match 'update(/:id)' **=>** 'pay\_rate#update'**,** :as **=>** :update**,** :via **=>** :post

match 'delete' **=>** 'pay\_rate#destroy'**,** :as **=>** :delete**,** :via **=>** :post

**end**

scope 'hourly'**,** :as **=>** 'hourly' **do**

match 'chart' **=>** 'hourly\_payroll\_chart#index'**,** :via **=>** :get

match 'chart/data' **=>** 'hourly\_payroll\_chart#data'**,** :via **=>** **[**:get**,** :post**]**

**end**

scope 'payslip'**,** :as **=>** 'payslip' **do**

match '' **=>** 'payslip#index'**,** :via **=>** :get

match 'list' **=>** 'payslip#list'**,** :via **=>** **[**:get**,** :post**]**

match 'slip(/:id(/:month(/:year)))' **=>** 'payslip#payslip'**,** :as **=>** :slip**,** :via **=>** :get

**end**

scope 'att'**,** :as **=>** 'att' **do**

match '' **=>** 'attendance#index'**,** :via **=>** :get

match 'list' **=>** 'attendance#list'**,** :via **=>** **[**:get**,** :post**]**

**end**

scope 'overtime'**,** :as **=>** 'overtime' **do**

scope 'rate'**,** :as **=>** 'rate' **do**

match '' **=>** 'overtime\_rate#index'**,** :via **=>** :get

match 'list' **=>** 'overtime\_rate#list'**,** :as **=>** :list**,** :via **=>** **[**:get**,** :post**]**

match 'new' **=>** 'overtime\_rate#new'**,** :as **=>** :new**,** :via **=>** :get

match 'create' **=>** 'overtime\_rate#create'**,** :as **=>** :create**,** :via **=>** :post

match 'edit(/:id)' **=>** 'overtime\_rate#edit'**,** :as **=>** :edit**,** :via **=>** :get

match 'update(/:id)' **=>** 'overtime\_rate#update'**,** :as **=>** :update**,** :via **=>** :post

match 'delete' **=>** 'overtime\_rate#destroy'**,** :as **=>** :delete**,** :via **=>** :post

**end**

match 'chart' **=>** 'overtime\_chart#index'**,** :via **=>** :get

match 'chart/data' **=>** 'overtime\_chart#data'**,** :via **=>** **[**:get**,** :post**]**

**end**

scope 'workhours'**,** :as **=>** 'workhours' **do**

match 'chart' **=>** 'total\_work\_hours\_chart#index'**,** :via **=>** :get

match 'chart/data' **=>** 'total\_work\_hours\_chart#data'**,** :via **=>** **[**:get**,** :post**]**

**end**

scope 'salaryadj'**,** :as **=>** 'salaryadj' **do**

match '' **=>** 'salary\_adjustment#index'**,** :via **=>** :get

match 'list' **=>** 'salary\_adjustment#list'**,** :as **=>** :list**,** :via **=>** **[**:get**,** :post**]**

match 'new' **=>** 'salary\_adjustment#new'**,** :as **=>** :new**,** :via **=>** :get

match 'create' **=>** 'salary\_adjustment#create'**,** :as **=>** :create**,** :via **=>** :post

match 'edit(/:id)' **=>** 'salary\_adjustment#edit'**,** :as **=>** :edit**,** :via **=>** :get

match 'update(/:id)' **=>** 'salary\_adjustment#update'**,** :as **=>** :update**,** :via **=>** :post

match 'delete' **=>** 'salary\_adjustment#destroy'**,** :as **=>** :delete**,** :via **=>** :post

**end**

**end**

namespace :user **do**

match 'index' **=>** 'user#index'**,** :as **=>** :index**,** :via **=>** :get

scope 'info'**,** :as **=>** 'info' **do**

match '' **=>** 'info#index'**,** :via **=>** :get

match 'update' **=>** 'info#update'**,** :via **=>** :post

**end**

scope 'contact'**,** :as **=>** 'contact' **do**

match '' **=>** 'contact#index'**,** :via **=>** :get

match 'update' **=>** 'contact#update'**,** :via **=>** :post

**end**

scope 'job'**,** :as **=>** 'job' **do**

match '' **=>** 'job#index'**,** :via **=>** :get

**end**

scope 'salary'**,** :as **=>** 'salary' **do**

match '' **=>** 'salary#index'**,** :via **=>** :get

**end**

scope 'qualification'**,** :as **=>** 'qualification' **do**

match '' **=>** 'qualification#index'**,** :via **=>** :get

match 'update' **=>** 'qualification#update'**,** :via **=>** :post

**end**

scope 'overtime'**,** :as **=>** 'overtime' **do**

match 'chart' **=>** 'overtime\_chart#index'**,** :via **=>** :get

match 'chart/data' **=>** 'overtime\_chart#data'**,** :via **=>** **[**:get**,** :post**]**

**end**

scope 'workhours'**,** :as **=>** 'workhours' **do**

match 'chart' **=>** 'total\_work\_hours\_chart#index'**,** :via **=>** :get

match 'chart/data' **=>** 'total\_work\_hours\_chart#data'**,** :via **=>** **[**:get**,** :post**]**

**end**

scope 'hourly'**,** :as **=>** 'hourly' **do**

match 'chart' **=>** 'hourly\_payroll\_chart#index'**,** :via **=>** :get

match 'chart/data' **=>** 'hourly\_payroll\_chart#data'**,** :via **=>** **[**:get**,** :post**]**

**end**

scope 'payslip'**,** :as **=>** 'payslip' **do**

match '' **=>** 'payslip#index'**,** :via **=>** :get

match 'slip(/:month(/:year))' **=>** 'payslip#payslip'**,** :as **=>** :slip**,** :via **=>** :get

**end**

**end**

# The priority is based upon order of creation:

# first created -> highest priority.

# Sample of regular route:

# match 'products/:id' => 'catalog#view'

# Keep in mind you can assign values other than :controller and :action

# Sample of named route:

# match 'products/:id/purchase' => 'catalog#purchase', :as => :purchase

# This route can be invoked with purchase\_url(:id => product.id)

# Sample resource route (maps HTTP verbs to controller actions automatically):

# resources :products

# Sample resource route with options:

# resources :products do

# member do

# get 'short'

# post 'toggle'

# end

#

# collection do

# get 'sold'

# end

# end

# Sample resource route with sub-resources:

# resources :products do

# resources :comments, :sales

# resource :seller

# end

# Sample resource route with more complex sub-resources

# resources :products do

# resources :comments

# resources :sales do

# get 'recent', :on => :collection

# end

# end

# Sample resource route within a namespace:

# namespace :admin do

# # Directs /admin/products/\* to Admin::ProductsController

# # (app/controllers/admin/products\_controller.rb)

# resources :products

# end

# You can have the root of your site routed with "root"

# just remember to delete public/index.html.

# root :to => 'welcome#index'

# See how all your routes lay out with "rake routes"

# This is a legacy wild controller route that's not recommended for RESTful applications.

# Note: This route will make all actions in every controller accessible via GET requests.

# match ':controller(/:action(/:id))(.:format)'

**end**

**E:\workspace\payroll\_rails\app\models\attendance.rb**

# Model for attendance table.

**class** **Attendance** **<** ActiveRecord**::**Base

attr\_accessible :id**,** :staff\_id**,** :time\_in**,** :time\_out**,** :work\_date

**self.**table\_name **=** 'attendance'

belongs\_to :employee**,** :foreign\_key **=>** 'staff\_id'**,** :primary\_key **=>** 'staff\_id'

validates\_presence\_of :work\_date**,** :message **=>** 'Work date is required'

validates\_presence\_of :time\_in**,** :message **=>** 'Time in is required'

validates\_presence\_of :time\_out**,** :message **=>** 'Time out is required'

**end**

**E:\workspace\payroll\_rails\app\models\department.rb**

# Model for department table.

**class** **Department** **<** ActiveRecord**::**Base

attr\_accessible :id**,** :name

**self.**table\_name **=** 'department'

validates\_presence\_of :name**,** :message **=>** 'Name is required'

validates\_uniqueness\_of :name**,** :message **=>** "Department %{value} already exist"

**end**

**E:\workspace\payroll\_rails\app\models\designation.rb**

# Model for designation table.

**class** **Designation** **<** ActiveRecord**::**Base

attr\_accessible :id**,** :title**,** :desc**,** :note

**self.**table\_name **=** 'designation'

has\_many :employee\_job

validates\_presence\_of :title**,** :message **=>** 'Job Title is required'

validates\_uniqueness\_of :title**,** :message **=>** "Job Title %{value} already exist"

**end**

**E:\workspace\payroll\_rails\app\models\employee.rb**

# Model for employee table.

**class** **Employee** **<** ActiveRecord**::**Base

attr\_accessible :dob**,** :staff\_id**,** :user\_id**,** :first\_name**,** :gender**,** :id**,** :is\_bumi**,**

:last\_name**,** :marital\_status**,** :middle\_name**,** :nationality**,** :new\_ic**,**

:old\_ic**,** :passport\_no**,** :place\_of\_birth**,** :race**,** :religion

**self.**table\_name **=** 'employee'

has\_one :employee\_contact**,** :foreign\_key **=>** 'id'

has\_one :employee\_job**,** :foreign\_key **=>** 'id'

has\_one :employee\_salary**,** :foreign\_key **=>** 'id'

has\_one :employee\_qualification**,** :foreign\_key **=>** 'id'

has\_one :employee**,** :foreign\_key **=>** 'id'

has\_many :attendance**,** :foreign\_key **=>** 'staff\_id'**,** :primary\_key **=>** 'staff\_id'

belongs\_to :user

validates\_presence\_of :staff\_id**,** :message **=>** 'Employee ID is required'

validates\_presence\_of :first\_name**,** :message **=>** 'First Name is required'

validates\_presence\_of :last\_name**,** :message **=>** 'Last Name is required'

validates\_presence\_of :new\_ic**,** :message **=>** 'New IC No. is required'

validates\_presence\_of :gender**,** :message **=>** 'Gender is required'

validates\_presence\_of :marital\_status**,** :message **=>** 'Marital Status is required'

validates\_presence\_of :nationality**,** :message **=>** 'Nationality is required'

validates\_presence\_of :dob**,** :message **=>** 'Date of Birth is required'

validates\_presence\_of :place\_of\_birth**,** :message **=>** 'Place of Birth is required'

validates\_presence\_of :race**,** :message **=>** 'Race is required'

validates\_uniqueness\_of :staff\_id**,** :message **=>** "Employee ID %{value} already exist"

**end**

**E:\workspace\payroll\_rails\app\models\employee\_contact.rb**

# Model for employee\_contact table.

**class** **EmployeeContact** **<** ActiveRecord**::**Base

attr\_accessible :address\_1**,** :address\_2**,** :address\_3**,** :city**,** :country**,** :home\_phone**,** :id**,**

:mobile\_phone**,** :other\_email**,** :postcode**,** :state**,** :work\_email

**self.**table\_name **=** 'employee\_contact'

belongs\_to :employee**,** :foreign\_key **=>** 'id'

validates\_presence\_of :address\_1**,** :message **=>** 'Address 1 is required'

validates\_presence\_of :city**,** :message **=>** 'City is required'

validates\_presence\_of :state**,** :message **=>** 'State is required'

validates\_presence\_of :postcode**,** :message **=>** 'Postal Code is required'

validates\_presence\_of :country**,** :message **=>** 'Country is required'

validates\_presence\_of :work\_email**,** :message **=>** 'Work Email is required'

**end**

**E:\workspace\payroll\_rails\app\models\employee\_job.rb**

# Model for employee\_job table.

**class** **EmployeeJob** **<** ActiveRecord**::**Base

attr\_accessible :confirm\_date**,** :department\_id**,** :designation\_id**,** :employment\_status\_id**,**

:id**,** :job\_category\_id**,** :join\_date

**self.**table\_name **=** 'employee\_job'

belongs\_to :employee**,** :foreign\_key **=>** 'id'

belongs\_to :designation

belongs\_to :department

belongs\_to :employment\_status

belongs\_to :job\_category

validates\_presence\_of :designation\_id**,** :message **=>** 'Designation is required'

validates\_presence\_of :department\_id**,** :message **=>** 'Department is required'

validates\_presence\_of :employment\_status\_id**,** :message **=>** 'Employment Status is required'

validates\_presence\_of :job\_category\_id**,** :message **=>** 'Job Category is required'

validates\_presence\_of :join\_date**,** :message **=>** 'Join Date is required'

**end**

**E:\workspace\payroll\_rails\app\models\employee\_qualification.rb**

# Model for employee\_qualification table.

**class** **EmployeeQualification** **<** ActiveRecord**::**Base

attr\_accessible :end\_date**,** :gpa**,** :id**,** :institute**,** :level**,** :major**,** :start\_date**,** :year

**self.**table\_name **=** 'employee\_qualification'

belongs\_to :employee**,** :foreign\_key **=>** 'id'

validates\_presence\_of :level**,** :message **=>** 'Qualification Level is required'

validates\_presence\_of :institute**,** :message **=>** 'Institute name is required'

validates\_presence\_of :year**,** :message **=>** 'Year obtained is required'

validates\_presence\_of :start\_date**,** :message **=>** 'Start Date is required'

validates\_presence\_of :end\_date**,** :message **=>** 'End Date is required'

**end**

**E:\workspace\payroll\_rails\app\models\employee\_salary.rb**

# Model for employee\_salary table.

**class** **EmployeeSalary** **<** ActiveRecord**::**Base

attr\_accessible :allowance**,** :bank\_acc\_no**,** :bank\_acc\_type**,** :bank\_address**,** :bank\_name**,**

:epf**,** :epf\_no**,** :id**,** :income\_tax**,** :income\_tax\_no**,** :salary**,** :socso\_no**,**

:pay\_type**,** :socso

**self.**table\_name **=** 'employee\_salary'

belongs\_to :employee**,** :foreign\_key **=>** 'id'

validates\_presence\_of :salary**,** :message **=>** 'Salary is required'

validates\_presence\_of :bank\_name**,** :message **=>** 'Bank Name is required'

validates\_presence\_of :bank\_acc\_no**,** :message **=>** 'Bank Account No. is required'

validates\_presence\_of :bank\_acc\_type**,** :message **=>** 'Bank Account Type is required'

validates\_presence\_of :bank\_address**,** :message **=>** 'Bank Address is required'

validates\_presence\_of :epf\_no**,** :message **=>** 'EPF No. is required'

validates\_presence\_of :pay\_type**,** :message **=>** 'Pay Type is required'

validates\_presence\_of :epf**,** :message **=>** 'EPF Deduction is required'

validates\_numericality\_of :salary**,** :greater\_than\_or\_equal\_to **=>** 0**,**

:message **=>** 'Salary is invalid'

validates\_numericality\_of :allowance**,** :greater\_than\_or\_equal\_to **=>** 0**,**

:message **=>** 'Allowance is invalid'

validates\_numericality\_of :epf**,** :greater\_than\_or\_equal\_to **=>** 0**,**

:message **=>** 'EPF Deduction is invalid'

validates\_numericality\_of :socso**,** :greater\_than\_or\_equal\_to **=>** 0**,**

:message **=>** 'SOCSO Deduction is invalid'

validates\_numericality\_of :income\_tax**,** :greater\_than\_or\_equal\_to **=>** 0**,**

:message **=>** 'Income Tax Deduction is invalid'

**def** **allowance**

a **=** read\_attribute**(**:allowance**)**

a**.**blank? **?** 0 : a

**end**

**def** **epf**

a **=** read\_attribute**(**:epf**)**

a**.**blank? **?** 0 : a

**end**

**def** **income\_tax**

a **=** read\_attribute**(**:income\_tax**)**

a**.**blank? **?** 0 : a

**end**

**def** **salary**

a **=** read\_attribute**(**:salary**)**

a**.**blank? **?** 0 : a

**end**

**def** **socso**

a **=** read\_attribute**(**:socso**)**

a**.**blank? **?** 0 : a

**end**

**def** **display\_pay\_type**

**case** **self.**pay\_type

**when** 1

'Monthly'

**when** 2

'Hourly'

**else**

''

**end**

**end**

**end**

**E:\workspace\payroll\_rails\app\models\employment\_status.rb**

# Model for employment\_status table.

**class** **EmploymentStatus** **<** ActiveRecord**::**Base

attr\_accessible :id**,** :name

**self.**table\_name **=** 'employment\_status'

has\_many :employee\_job

validates\_presence\_of :name**,** :message **=>** 'Name is required'

validates\_uniqueness\_of :name**,** :message **=>** "Employment Status %{value} already exist"

**end**

**E:\workspace\payroll\_rails\app\models\job\_category.rb**

# Model for job\_category table.

**class** **JobCategory** **<** ActiveRecord**::**Base

attr\_accessible :id**,** :name

**self.**table\_name **=** 'job\_category'

has\_many :employee\_job

validates\_presence\_of :name**,** :message **=>** 'Name is required'

validates\_uniqueness\_of :name**,** :message **=>** "Category %{value} already exist"

**end**

**E:\workspace\payroll\_rails\app\models\overtime\_rate.rb**

# Model for overtime\_rate table.

**class** **OvertimeRate** **<** ActiveRecord**::**Base

attr\_accessible :duration**,** :pay\_rate**,** :year

**self.**table\_name **=** 'overtime\_rate'

validates\_presence\_of :duration**,** :message **=>** 'Duration is required'

validates\_presence\_of :year**,** :message **=>** 'Year is required'

validates\_presence\_of :pay\_rate**,** :message **=>** 'Pay Rate is required'

validates\_numericality\_of :duration**,** :greater\_than\_or\_equal\_to **=>** 0**,**

:message **=>** 'Duration is invalid'

validates\_numericality\_of :year**,** :greater\_than **=>** 0**,** :message **=>** 'Year is invalid'

validates\_numericality\_of :pay\_rate**,** :greater\_than **=>** 0**,**

:message **=>** 'Pay Rate is invalid'

validates\_uniqueness\_of :year**,**

:message **=>** "Overtime rate for year %{value} already exist"

**def** **duration**

a **=** read\_attribute**(**:duration**)**

a**.**blank? **?** 0 : a

**end**

**def** **pay\_rate**

a **=** read\_attribute**(**:pay\_rate**)**

a**.**blank? **?** 0 : a

**end**

**end**

**E:\workspace\payroll\_rails\app\models\pay\_rate.rb**

# Model for pay\_rate table.

**class** **PayRate** **<** ActiveRecord**::**Base

attr\_accessible :hourly\_pay\_rate**,** :id**,** :month**,** :staff\_id**,** :year

**self.**table\_name **=** 'pay\_rate'

validates\_presence\_of :staff\_id**,** :message **=>** 'Staff ID is required'

validates\_presence\_of :month**,** :message **=>** 'Month is required'

validates\_presence\_of :year**,** :message **=>** 'Year is required'

validates\_presence\_of :hourly\_pay\_rate**,** :message **=>** 'Hourly pay rate is required'

validates\_numericality\_of :month**,** :greater\_than **=>** 0**,** :message **=>** 'Month is invalid'

validates\_numericality\_of :month**,** :less\_than\_or\_equal\_to **=>** 12**,**

:message **=>** 'Month is invalid'

validates\_numericality\_of :year**,** :greater\_than **=>** 0**,** :message **=>** 'Year is invalid'

validates\_numericality\_of :hourly\_pay\_rate**,** :greater\_than **=>** 0**,**

:message **=>** 'Hourly pay rate is invalid'

**def** **hourly\_pay\_rate**

a **=** read\_attribute**(**:hourly\_pay\_rate**)**

a**.**blank? **?** 0 : a

**end**

**end**

**E:\workspace\payroll\_rails\app\models\salary\_adjustment.rb**

# Model for salary\_adjustment table.

**class** **SalaryAdjustment** **<** ActiveRecord**::**Base

attr\_accessible :id**,** :inc**,** :month**,** :staff\_id**,** :year

**self.**table\_name **=** 'salary\_adjustment'

validates\_presence\_of :staff\_id**,** :message **=>** 'Staff ID is required'

validates\_presence\_of :inc**,** :message **=>** 'Increment is required'

validates\_presence\_of :month**,** :message **=>** 'Month is required'

validates\_presence\_of :year**,** :message **=>** 'Year is required'

validates\_numericality\_of :inc**,** :greater\_than **=>** 0**,**

:message **=>** 'Increment is invalid'

validates\_numericality\_of :month**,** :greater\_than **=>** 0**,** :message **=>** 'Month is invalid'

validates\_numericality\_of :month**,** :less\_than\_or\_equal\_to **=>** 12**,**

:message **=>** 'Month is invalid'

validates\_numericality\_of :year**,** :greater\_than **=>** 0**,** :message **=>** 'Year is invalid'

**def** **inc**

a **=** read\_attribute**(**:inc**)**

a**.**blank? **?** 0 : a

**end**

**end**

**E:\workspace\payroll\_rails\app\models\user.rb**

require 'digest'

# Model for user table.

**class** **User** **<** ActiveRecord**::**Base

attr\_accessible :id**,** :pwd**,** :role**,** :status**,** :username**,** :pwd\_confirmation

attr\_accessor :pwd

**self.**table\_name **=** 'user'

has\_one :employee

validates :username**,** :uniqueness **=>** **{** :message **=>** "Username %{value} already exist" **},**

:length **=>** **{** :within **=>** 3**..**50**,**

:message **=>** "Minimum is %{count} characters" **}**

validates :pwd**,** :confirmation **=>** **{** :message **=>** "Password doesn't match confirmation" **},**

:length **=>** **{** :within **=>** 4**..**20**,**

:message **=>** "Minimum is %{count} characters" **},**

:presence **=>** **{** :message **=>** "Password is required" **},**

:if **=>** :password\_required**?**

before\_save :encrypt\_new\_password

UNCHANGED\_PASSWORD **=** '\*\*\*\*\*\*\*\*'

ADMIN **=** 1

NORMAL\_USER **=** 2

@@roles **=** **{** 'Admin' **=>** 1**,** 'Normal User' **=>** 2 **}**

@@statuses **=** **{** 'Enabled' **=>** 1**,** 'Disabled' **=>** 0 **}**

**def** **self.**authenticate**(**username**,** password**)**

user **=** find\_by\_username**(**username**)**

**if** user **&&** user**.**authenticated?**(**password**)** **&&** user**.**enabled?

**return** user

**end**

**end**

**def** **authenticated?(**password**)**

**self.**password **==** encrypt**(**password**)**

**end**

**def** **enabled?**

**self.**status **==** **true**

**end**

**def** **self.**roles

@@roles

**end**

**def** **self.**statuses

@@statuses

**end**

**def** **role\_display**

role **==** 1 **?** 'Admin' : 'Normal User'

**end**

**def** **status\_display**

status **==** **true** **?** 'Enabled' : 'Disabled'

**end**

protected

**def** **encrypt\_new\_password**

**return** if pwd**.**blank?

**self.**password **=** encrypt**(**pwd**)**

**end**

**def** **password\_required?**

**if** pwd **==** UNCHANGED\_PASSWORD

**return** **false**

**end**

**self.**password**.**blank? **||** pwd**.**present?

**end**

**def** **encrypt(**string**)**

Digest**::**SHA1**.**hexdigest**(**string**)**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\application\_controller.rb**

# The main application's controller.

**class** **ApplicationController** **<** ActionController**::**Base

protect\_from\_forgery

LAYOUT **=** **{**

:admin **=>** 'admin'**,**

:chart **=>** 'chart'**,**

:list **=>** 'list'**,**

:user **=>** 'user'

**}**

# Authenticates a user.

**def** **create**

user **=** User**.**authenticate**(**params**[**:username**],** params**[**:password**])**

**if** user**.**present?

session**[**:user\_id**]** **=** user**.**id

**if** user**.**role **==** User**::**ADMIN

redirect\_to admin\_index\_path **and** **return**

**else**

employee **=** user**.**employee

**if** employee**.**present?

session**[**:employee\_id**]** **=** employee**.**id

session**[**:staff\_id**]** **=** employee**.**staff\_id

session**[**:supervisor\_id**]** **=** employee**.**id

redirect\_to user\_index\_path **and** **return**

**else**

flash**.**now**[**:alert**]** **=** %Q{No employee record found. Please contact the administrator

to create your employee record.}

**end**

**end**

**else**

flash**.**now**[**:alert**]** **=** 'Incorrect username or password'

**end**

render :action **=>** 'new'

**end**

# Logs out a user.

**def** **destroy**

reset\_session

redirect\_to login\_path

**end**

protected

# Returns the current user's user\_id.

**def** **current\_user**

**return** unless session**[**:user\_id**]**

@current\_user **||=** session**[**:user\_id**]**

**end**

# Checks whether a user is authenticated.

**def** **authenticate**

logged\_in? **?** **true** : access\_denied

**end**

# Checks whether a user is logged in.

**def** **logged\_in?**

current\_user**.**present?

**end**

# Prevents a user from accessing the system without logging in.

**def** **access\_denied**

redirect\_to login\_path**,** :notice **=>** 'Please log in to continue' **and** **return** **false**

**end**

# Formats a date into dd-mm-yyyy .

# Helper method.

**def** **fmt\_date(**dt**)**

**if** dt**.**present?

dt**.**strftime**(**'%d-%m-%Y'**)**

**end**

**end**

# Formats a time into hh:mm:ss AM/PM .

# Helper method.

**def** **fmt\_time(**t**)**

**if** t**.**present?

t**.**in\_time\_zone**(**'Kuala Lumpur'**).**strftime**(**'%l:%M %p'**)**

**end**

**end**

# Returns the month name from a given number.

# Helper method.

**def** **month\_name(**i**)**

ApplicationHelper**.**month\_name**(**i**)**

**end**

# Returns a hash of month names.

**def** **month\_options**

months **=** I18n**.**t**(**'date.month\_names'**)**

o **=** **{}**

**(**1**..**12**).**each **do** **|**m**|**

o**[**months**[**m**]]** **=** m

**end**

o

**end**

# Returns the end year for year selection in view.

# Helper method.

**def** **end\_year**

2000

**end**

# Returns the user\_id.

**def** **get\_user\_id**

session**[**:user\_id**]**

**end**

helper\_method :current\_user

helper\_method :logged\_in

helper\_method :fmt\_date

helper\_method :fmt\_time

helper\_method :month\_name

helper\_method :end\_year

**end**

**E:\workspace\payroll\_rails\app\controllers\admin\admin\_controller.rb**

# Admin base controller.

# All Admin controllers must inherit this controller.

# Admin controller serves incoming requests from an authenticated Admin User.

**class** **Admin::**AdminController **<** ApplicationController

layout **false**

# checks whether a user is authenticated before serving any request

before\_filter :authenticate

# Display the admin page.

# GET /admin/index

**def** **index**

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:admin**]** **}**

**end**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\admin\attendance\_controller.rb**

# This controller serves incoming requests to display out the Attendance records.

**class** **Admin::**AttendanceController **<** Admin**::**AdminController

# List all records.

# GET /att

# GET /att.json

**def** **index**

@data **=** AttendanceHelper**.**get\_all

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:list**]** **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# List records by filtering.

# GET /att/list

# GET /att/list.json

**def** **list**

\_work\_date **=** params**[**:work\_date**]**

employee **=** params**[**:employee**].**blank? **?** '' : params**[**:employee**]**

work\_date **=** Date**.**strptime**(**\_work\_date**,** '%d-%m-%Y'**)** if \_work\_date**.**present?

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

sortcolumn **=** params**[**:sortcolumn**].**blank? **?** AttendanceHelper**::**DEFAULT\_SORT\_COLUMN

**:** params**[**:sortcolumn**]**

sortdir **=** params**[**:sortdir**].**blank? **?** AttendanceHelper**::**DEFAULT\_SORT\_DIR :

params**[**:sortdir**]**

sort **=** ApplicationHelper**::**Sort**.**new**(**sortcolumn**,** sortdir**)**

filters **=** **{** :work\_date **=>** work\_date**,**

:employee **=>** employee **}**

**if** work\_date**.**blank? **&&** employee**.**blank?

@data **=** AttendanceHelper**.**get\_all**(**pgnum**,** pgsize**,** sort**)**

**else**

@data **=** AttendanceHelper**.**get\_filter\_by**(**filters**,** pgnum**,** pgsize**,** sort**)**

**end**

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'list' **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\admin\department\_controller.rb**

# This controller serves incoming requests to display out the Department records.

**class** **Admin::**DepartmentController **<** Admin**::**AdminController

# List all records.

# GET /dept

# GET /dept.json

**def** **index**

@data **=** DepartmentHelper**.**get\_all

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:list**]** **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# List records by filtering.

# GET /dept/list

# GET /dept/list.json

**def** **list**

keyword **=** params**[**:keyword**].**blank? **?** '' : params**[**:keyword**]**

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

sortcolumn **=** params**[**:sortcolumn**].**blank? **?** DepartmentHelper**::**DEFAULT\_SORT\_COLUMN

**:** params**[**:sortcolumn**]**

sortdir **=** params**[**:sortdir**].**blank? **?** DepartmentHelper**::**DEFAULT\_SORT\_DIR :

params**[**:sortdir**]**

sort **=** ApplicationHelper**::**Sort**.**new**(**sortcolumn**,** sortdir**)**

**if** keyword**.**blank?

@data **=** DepartmentHelper**.**get\_all**(**pgnum**,** pgsize**,** sort**)**

**else**

@data **=** DepartmentHelper**.**get\_filter\_by**(**keyword**,** pgnum**,** pgsize**,** sort**)**

**end**

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'list' **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# Display the Department form.

# GET /dept/new

# GET /dept/new.json

**def** **new**

@dept **=** Department**.**new

@form\_id **=** 'add-form'

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @dept **}**

**end**

**end**

# Create new Department record.

# POST /dept/create

**def** **create**

o **=** Department**.**new**(**:name **=>** params**[**:name**])**

**if** o**.**save

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Department was successfully added.' **}**

**else**

render :json **=>** DepartmentHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

# Display the Department form, with existing record to edit.

# GET /dept/edit/1

# GET /dept/edit/1.json

**def** **edit**

@dept **=** Department**.**find**(**params**[**:id**])**

@form\_id **=** 'edit-form'

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @dept **}**

**end**

**end**

# Update Department record.

# POST /dept/update/1

**def** **update**

o **=** Department**.**find**(**params**[**:id**])**

**if** o**.**update\_attributes**(**:name **=>** params**[**:name**])**

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Department was successfully updated.' **}**

**else**

render :json **=>** DepartmentHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

# Delete a list of Department records.

# POST /dept/delete

**def** **destroy**

keyword **=** params**[**:keyword**].**blank? **?** '' : params**[**:keyword**]**

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

ids **=** params**[**:id**]**

Department**.**delete\_all**(**:id **=>** ids**)**

itemscount **=** DepartmentHelper**.**item\_message**(**keyword**,** pgnum**,** pgsize**)**

render :json **=>** **{** :success **=>** 1**,**

:itemscount **=>** itemscount**,**

:message **=>** "**#{**ids**.**size**}** Department(s) was successfully deleted." **}**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\admin\designation\_controller.rb**

# This controller serves incoming requests to display out the Designation records.

**class** **Admin::**DesignationController **<** Admin**::**AdminController

# List all records.

# GET /designation

# GET /designation.json

**def** **index**

@data **=** DesignationHelper**.**get\_all

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:list**]** **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# List records by filtering.

# GET /designation/list

# GET /designation/list.json

**def** **list**

find **=** params**[**:find**].**blank? **?** 0 : params**[**:find**].**to\_i

keyword **=** params**[**:keyword**].**blank? **?** '' : params**[**:keyword**]**

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

sortcolumn **=** params**[**:sortcolumn**].**blank? **?** DesignationHelper**::**DEFAULT\_SORT\_COLUMN

**:** params**[**:sortcolumn**]**

sortdir **=** params**[**:sortdir**].**blank? **?** DesignationHelper**::**DEFAULT\_SORT\_DIR :

params**[**:sortdir**]**

sort **=** ApplicationHelper**::**Sort**.**new**(**sortcolumn**,** sortdir**)**

**if** find **==** 0 **&&** keyword**.**blank?

@data **=** DesignationHelper**.**get\_all**(**pgnum**,** pgsize**,** sort**)**

**else**

@data **=** DesignationHelper**.**get\_filter\_by**(**find**,** keyword**,** pgnum**,** pgsize**,** sort**)**

**end**

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'list' **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# Display the Designation form.

# GET /designation/new

# GET /designation/new.json

**def** **new**

@designation **=** Designation**.**new

@form\_id **=** 'add-form'

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @designation **}**

**end**

**end**

# Create new Designation record.

# POST /designation/create

**def** **create**

o **=** Designation**.**new**(**:title **=>** params**[**:title**],** :desc **=>** params**[**:desc**],**

:note **=>** params**[**:note**])**

**if** o**.**save

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Job Title was successfully added.' **}**

**else**

render :json **=>** DesignationHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

# Display the Designation form, with existing record to edit.

# GET /designation/edit/1

# GET /designation/edit/1.json

**def** **edit**

@designation **=** Designation**.**find**(**params**[**:id**])**

@form\_id **=** 'edit-form'

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @designation **}**

**end**

**end**

# Update Designation record.

# POST /designation/update/1

**def** **update**

o **=** Designation**.**find**(**params**[**:id**])**

**if** o**.**update\_attributes**(**:title **=>** params**[**:title**],** :desc **=>** params**[**:desc**],**

:note **=>** params**[**:note**])**

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Job Title was successfully updated.' **}**

**else**

render :json **=>** DesignationHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

# Delete a list of Designation records.

# POST /designation/delete

**def** **destroy**

keyword **=** params**[**:keyword**].**blank? **?** '' : params**[**:keyword**]**

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

ids **=** params**[**:id**]**

Designation**.**delete\_all**(**:id **=>** ids**)**

itemscount **=** DesignationHelper**.**item\_message**(**keyword**,** pgnum**,** pgsize**)**

render :json **=>** **{** :success **=>** 1**,**

:itemscount **=>** itemscount**,**

:message **=>** "**#{**ids**.**size**}** Job Title(s) was successfully deleted." **}**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\admin\employee\_controller.rb**

require 'securerandom'

# This controller serves incoming requests to display out the Employee records.

**class** **Admin::**EmployeeController **<** Admin**::**AdminController

# List all records.

# GET /employee

# GET /employee.json

**def** **index**

@data **=** EmployeeHelper**.**get\_all

@employmentstatus **=** EmploymentStatus**.**order**(**:name**).**all

@designation **=** Designation**.**order**(**:title**).**all

@dept **=** Department**.**order**(**:name**).**all

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:list**]** **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# List records by filtering.

# GET /employee/list

# GET /employee/list.json

**def** **list**

employee **=** params**[**:employee**].**blank? **?** '' : params**[**:employee**]**

staff\_id **=** params**[**:staff\_id**].**blank? **?** '' : params**[**:staff\_id**]**

employment\_status **=** params**[**:employment\_status**].**blank? **?** 0 :

params**[**:employment\_status**].**to\_i

designation **=** params**[**:designation**].**blank? **?** 0 : params**[**:designation**].**to\_i

dept **=** params**[**:dept**].**blank? **?** 0 : params**[**:dept**].**to\_i

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

sortcolumn **=** params**[**:sortcolumn**].**blank? **?** EmployeeHelper**::**DEFAULT\_SORT\_COLUMN

**:** params**[**:sortcolumn**]**

sortdir **=** params**[**:sortdir**].**blank? **?** EmployeeHelper**::**DEFAULT\_SORT\_DIR :

params**[**:sortdir**]**

sort **=** ApplicationHelper**::**Sort**.**new**(**sortcolumn**,** sortdir**)**

filters **=** **{** :employee **=>** employee**,**

:staff\_id **=>** staff\_id**,**

:employment\_status **=>** employment\_status**,**

:designation **=>** designation**,**

:dept **=>** dept **}**

**if** employee**.**blank? **&&** staff\_id**.**blank? **&&** employment\_status **==** 0 **&&** designation **==** 0 **&&**

dept **==** 0

@data **=** EmployeeHelper**.**get\_all**(**pgnum**,** pgsize**,** sort**)**

**else**

@data **=** EmployeeHelper**.**get\_filter\_by**(**filters**,** pgnum**,** pgsize**,** sort**)**

**end**

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'list' **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# Display the Employee form.

# GET /employee/new

# GET /employee/new.json

**def** **new**

@employee **=** Employee**.**new

@employee\_contact **=** EmployeeContact**.**new

@employee\_job **=** EmployeeJob**.**new

@employee\_salary **=** EmployeeSalary**.**new

@employee\_qualification **=** EmployeeQualification**.**new

@form\_id **=** 'add-form'

@users **=** User**.**order**(**:username**).**all

@designations **=** Designation**.**order**(**:title**).**all

@employment\_statuses **=** EmploymentStatus**.**order**(**:name**).**all

@job\_categories **=** JobCategory**.**order**(**:name**).**all

@departments **=** Department**.**order**(**:name**).**all

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @employee **}**

**end**

**end**

# Create new Employee record.

# POST /employee/create

**def** **create**

o **=** EmployeeHelper**.**employee\_obj**(**params**)**

b1 **=** EmployeeContactHelper**.**is\_empty\_params?**(**params**)**

oc **=** EmployeeContactHelper**.**employee\_contact\_obj**(**o**,** params**)**

b2 **=** EmployeeJobHelper**.**is\_empty\_params?**(**params**)**

oej **=** EmployeeJobHelper**.**employee\_job\_obj**(**o**,** params**)**

b3 **=** EmployeeSalaryHelper**.**is\_empty\_params?**(**params**)**

osa **=** EmployeeSalaryHelper**.**employee\_salary\_obj**(**o**,** params**)**

b4 **=** EmployeeQualificationHelper**.**is\_empty\_params?**(**params**)**

oq **=** EmployeeQualificationHelper**.**employee\_qualification\_obj**(**o**,** params**)**

v1 **=** o**.**valid?

v2 **=** b1 **?** **true** : oc**.**valid?

v3 **=** b2 **?** **true** : oej**.**valid?

v4 **=** b3 **?** **true** : osa**.**valid?

v5 **=** b4 **?** **true** : oq**.**valid?

**if** **!**v1 **||** **!**v2 **||** **!**v3 **||** **!**v4 **||** **!**v5

employee\_errors **=** EmployeeHelper**.**get\_errors**(**o**.**errors**)**

employee\_contact\_errors **=** EmployeeContactHelper**.**get\_errors**(**oc**.**errors**)**

employee\_job\_errors **=** EmployeeJobHelper**.**get\_errors**(**oej**.**errors**)**

employee\_salary\_errors **=** EmployeeSalaryHelper**.**get\_errors**(**osa**.**errors**)**

employee\_qualification\_errors **=** EmployeeQualificationHelper**.**get\_errors**(**oq**.**errors**)**

errors **=** **{** :error **=>** 1**,** :employee **=>** employee\_errors**,**

:employee\_contact **=>** employee\_contact\_errors**,**

:employee\_job **=>** employee\_job\_errors**,**

:employee\_salary **=>** employee\_salary\_errors**,**

:employee\_qualification **=>** employee\_qualification\_errors **}**

render :json **=>** errors **and** **return**

**end**

ActiveRecord**::**Base**.**transaction **do**

o**.**save

oc**.**save if b1

oej**.**save if b2

osa**.**save if b3

oq**.**save if b4

**end**

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Employee was successfully added.' **}**

**end**

# Display the Employee form, with existing record to edit.

# GET /employee/edit/1

# GET /employee/edit/1.json

**def** **edit**

@employee **=** Employee**.**find**(**params**[**:id**])**

@employee\_contact **=** @employee**.**employee\_contact**.**blank? **?** EmployeeContact**.**new

**:** @employee**.**employee\_contact

@employee\_job **=** @employee**.**employee\_job**.**blank? **?** EmployeeJob**.**new

**:** @employee**.**employee\_job

@employee\_salary **=** @employee**.**employee\_salary**.**blank? **?** EmployeeSalary**.**new

**:** @employee**.**employee\_salary

@employee\_qualification **=** @employee**.**employee\_qualification**.**blank? **?**

EmployeeQualification**.**new : @employee**.**employee\_qualification

@form\_id **=** 'edit-form'

@users **=** User**.**order**(**:username**).**all

@designations **=** Designation**.**order**(**:title**).**all

@employment\_statuses **=** EmploymentStatus**.**order**(**:name**).**all

@job\_categories **=** JobCategory**.**order**(**:name**).**all

@departments **=** Department**.**order**(**:name**).**all

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @employee **}**

**end**

**end**

# Update Employee record.

# POST /employee/update

**def** **update**

o **=** Employee**.**find**(**params**[**:id**])**

oc **=** o**.**employee\_contact

oej **=** o**.**employee\_job

osa **=** o**.**employee\_salary

oq **=** o**.**employee\_qualification

oc\_new **=** **false**

**if** oc**.**blank?

oc **=** EmployeeContactHelper**.**employee\_contact\_obj**(**o**,** params**)**

oc\_new **=** **true**

**end**

oej\_new **=** **false**

**if** oej**.**blank?

oej **=** EmployeeJobHelper**.**employee\_job\_obj**(**o**,** params**)**

oej\_new **=** **true**

**end**

osa\_new **=** **false**

**if** osa**.**blank?

osa **=** EmployeeSalaryHelper**.**employee\_salary\_obj**(**o**,** params**)**

osa\_new **=** **true**

**end**

oq\_new **=** **false**

**if** oq**.**blank?

oq **=** EmployeeQualificationHelper**.**employee\_qualification\_obj**(**o**,** params**)**

oq\_new **=** **true**

**end**

b1 **=** EmployeeContactHelper**.**is\_empty\_params?**(**params**)**

b2 **=** EmployeeJobHelper**.**is\_empty\_params?**(**params**)**

b3 **=** EmployeeSalaryHelper**.**is\_empty\_params?**(**params**)**

b4 **=** EmployeeQualificationHelper**.**is\_empty\_params?**(**params**)**

v1 **=** o**.**valid?

v2 **=** b1 **?** **true** : oc**.**valid?

v3 **=** b2 **?** **true** : oej**.**valid?

v4 **=** b3 **?** **true** : osa**.**valid?

v5 **=** b4 **?** **true** : oq**.**valid?

**if** **!**v1 **||** **!**v2 **||** **!**v3 **||** **!**v4 **||** **!**v5

employee\_errors **=** EmployeeHelper**.**get\_errors**(**o**.**errors**)**

employee\_contact\_errors **=** EmployeeContactHelper**.**get\_errors**(**oc**.**errors**)**

employee\_job\_errors **=** EmployeeJobHelper**.**get\_errors**(**oej**.**errors**)**

employee\_salary\_errors **=** EmployeeSalaryHelper**.**get\_errors**(**osa**.**errors**)**

employee\_qualification\_errors **=** EmployeeQualificationHelper**.**get\_errors**(**oq**.**errors**)**

errors **=** **{** :error **=>** 1**,** :employee **=>** employee\_errors**,**

:employee\_contact **=>** employee\_contact\_errors**,**

:employee\_job **=>** employee\_job\_errors**,**

:employee\_salary **=>** employee\_salary\_errors**,**

:employee\_qualification **=>** employee\_qualification\_errors **}**

render :json **=>** errors **and** **return**

**end**

ActiveRecord**::**Base**.**transaction **do**

EmployeeHelper**.**update\_obj**(**o**,** params**)**

**if** oc\_new

oc**.**save

**else**

EmployeeContactHelper**.**update\_obj**(**oc**,** params**)**

**end**

**if** oej\_new

oej**.**save

**else**

EmployeeJobHelper**.**update\_obj**(**oej**,** params**)**

**end**

**if** osa\_new

osa**.**save

**else**

EmployeeSalaryHelper**.**update\_obj**(**osa**,** params**)**

**end**

**if** oq\_new

oq**.**save

**else**

EmployeeQualificationHelper**.**update\_obj**(**oq**,** params**)**

**end**

**end**

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Employee was successfully updated.' **}**

**end**

# Delete a list of Employee records.

# POST /employee/delete

**def** **destroy**

employee **=** params**[**:employee**].**blank? **?** '' : params**[**:employee**]**

staff\_id **=** params**[**:staff\_id**].**blank? **?** '' : params**[**:staff\_id**]**

employment\_status **=** params**[**:employment\_status**].**blank? **?** 0 : params**[**:employment\_status**].**to\_i

designation **=** params**[**:designation**].**blank? **?** 0 : params**[**:designation**].**to\_i

dept **=** params**[**:dept**].**blank? **?** 0 : params**[**:dept**].**to\_i

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

ids **=** params**[**:id**]**

Employee**.**delete\_all**(**:id **=>** ids**)**

filters **=** **{** :employee **=>** employee**,**

:staff\_id **=>** staff\_id**,**

:employment\_status **=>** employment\_status**,**

:designation **=>** designation**,**

:dept **=>** dept **}**

**if** employee**.**blank? **&&** staff\_id**.**blank? **&&** employment\_status **==** 0 **&&** designation **==** 0 **&&**

dept **==** 0

itemscount **=** EmployeeHelper**.**item\_message**(nil,** pgnum**,** pgsize**)**

**else**

itemscount **=** EmployeeHelper**.**item\_message**(**filters**,** pgnum**,** pgsize**)**

**end**

render :json **=>** **{** :success **=>** 1**,**

:itemscount **=>** itemscount**,**

:message **=>** "**#{**ids**.**size**}** employee(s) was successfully deleted." **}**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\admin\employment\_status\_controller.rb**

# This controller serves incoming requests to display out the EmploymentStatus records.

**class** **Admin::**EmploymentStatusController **<** Admin**::**AdminController

# List all records.

# GET /empstatus

# GET /empstatus.json

**def** **index**

@data **=** EmploymentStatusHelper**.**get\_all

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:list**]** **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# List records by filtering.

# GET /empstatus/list

# GET /empstatus/list.json

**def** **list**

keyword **=** params**[**:keyword**].**blank? **?** '' : params**[**:keyword**]**

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

sortcolumn **=** params**[**:sortcolumn**].**blank? **?** EmploymentStatusHelper**::**DEFAULT\_SORT\_COLUMN

**:** params**[**:sortcolumn**]**

sortdir **=** params**[**:sortdir**].**blank? **?** EmploymentStatusHelper**::**DEFAULT\_SORT\_DIR

**:** params**[**:sortdir**]**

sort **=** ApplicationHelper**::**Sort**.**new**(**sortcolumn**,** sortdir**)**

**if** keyword**.**blank?

@data **=** EmploymentStatusHelper**.**get\_all**(**pgnum**,** pgsize**,** sort**)**

**else**

@data **=** EmploymentStatusHelper**.**get\_filter\_by**(**keyword**,** pgnum**,** pgsize**,** sort**)**

**end**

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'list' **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# Display the EmploymentStatus form.

# GET /empstatus/new

# GET /empstatus/new.json

**def** **new**

@empstatus **=** EmploymentStatus**.**new

@form\_id **=** 'add-form'

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @empstatus **}**

**end**

**end**

# Create new EmploymentStatus record.

# POST /empstatus/create

**def** **create**

o **=** EmploymentStatus**.**new**(**:name **=>** params**[**:name**])**

**if** o**.**save

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Employment Status was successfully added.' **}**

**else**

render :json **=>** EmploymentStatusHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

# Display the EmploymentStatus form, with existing record to edit.

# GET /empstatus/edit/1

# GET /empstatus/edit/1.json

**def** **edit**

@empstatus **=** EmploymentStatus**.**find**(**params**[**:id**])**

@form\_id **=** 'edit-form'

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @empstatus **}**

**end**

**end**

# Update EmploymentStatus record.

# POST /empstatus/update/1

**def** **update**

o **=** EmploymentStatus**.**find**(**params**[**:id**])**

**if** o**.**update\_attributes**(**:name **=>** params**[**:name**])**

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Employment Status was successfully updated.' **}**

**else**

render :json **=>** EmploymentStatusHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

# Delete a list of EmploymentStatus records.

# POST /empstatus/delete

**def** **destroy**

keyword **=** params**[**:keyword**].**blank? **?** '' : params**[**:keyword**]**

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

ids **=** params**[**:id**]**

EmploymentStatus**.**delete\_all**(**:id **=>** ids**)**

itemscount **=** EmploymentStatusHelper**.**item\_message**(**keyword**,** pgnum**,** pgsize**)**

render :json **=>** **{** :success **=>** 1**,**

:itemscount **=>** itemscount**,**

:message **=>** %Q{**#{**ids**.**size**}** Employment Status(es) was successfully

deleted.} **}**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\admin\hourly\_payroll\_chart\_controller.rb**

# This controller serves incoming requests to display Hourly Payroll Chart.

**class** **Admin::**HourlyPayrollChartController **<** Admin**::**AdminController

# Display the main page.

# GET /hourly/chart

**def** **index**

@month\_hash **=** month\_options

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:chart**]** **}**

**end**

**end**

# Get the hourly payroll json data to be populated to the chart.

# GET /hourly/chart/data

**def** **data**

staff\_id **=** params**[**:staff\_id**].**blank? **?** '' : params**[**:staff\_id**]**

month **=** params**[**:month**].**blank? **?** '0' : params**[**:month**]**

year **=** params**[**:year**].**blank? **?** 0 : params**[**:year**].**to\_i

title **=** 'Hourly Payroll'

yaxis **=** 'Total Amount (RM)'

months **=** I18n**.**t**(**'date.month\_names'**)**

o **=** Array**.**new**(**12**)** **{** **|**x**|** **[**months**[**x **+** 1**],** 0**]** **}**

b **=** Array**.**new**(**12**)** **{** **|**x**|** 0 **}**

categories **=** Array**.**new**(**12**)** **{** **|**x**|** months**[**x **+** 1**][**0**..**2**]** **}**

**if** staff\_id**.**present?

liststaff **=** **[**staff\_id**]**

**else**

list **=** PayRate**.**select**(**'distinct(staff\_id)'**).**all

liststaff **=** list**.**collect **{** **|**x**|** x**.**staff\_id **}**

**end**

**if** year **!=** 0

listyear **=** **[**year**]**

title **=** "Hourly Payroll for **#{**year**}**"

**else**

list **=** PayRate**.**select**(**'distinct(year)'**).**all

listyear **=** list**.**collect **{** **|**x**|** x**.**year **}**

**end**

**if** month **!=** '0'

listmonth **=** month**.**collect **{** **|**x**|** x**.**to\_i **}**

**else**

list **=** PayRate**.**select**(**'distinct(month)'**).**all

listmonth **=** list**.**collect **{** **|**x**|** x**.**month **}**

**end**

listyear**.**each **do** **|**y**|**

listmonth**.**each **do** **|**m**|**

liststaff**.**each **do** **|**s**|**

filters **=** **{** :year **=>** y**,** :month **=>** m**,** :staff\_id **=>** s **}**

total\_hours **=** AttendanceHelper**.**get\_total\_hours**(**filters**)**

rate **=** PayRateHelper**.**get\_pay\_rate**(**filters**)**

v **=** total\_hours **\*** rate

o**[**m **-** 1**][**1**]** **=** v

b**[**m **-** 1**]** **+=** v

**end**

**end**

**end**

c **=** b**.**collect **{** **|**x**|** x**.**round**(**2**)** **}**

@data **=** **{** :pie **=>** o**,** :column **=>** **{** :data **=>** c**,**

:categories **=>** categories**,**

:yaxis **=>** yaxis **},**

:title **=>** title **}**

render :json **=>** @data

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\admin\job\_category\_controller.rb**

# This controller serves incoming requests to display out the JobCategory records.

**class** **Admin::**JobCategoryController **<** Admin**::**AdminController

# List all records.

# GET /jobcat

# GET /jobcat.json

**def** **index**

@data **=** JobCategoryHelper**.**get\_all

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:list**]** **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# List records by filtering.

# GET /jobcat/list

# GET /jobcat/list.json

**def** **list**

keyword **=** params**[**:keyword**].**blank? **?** '' : params**[**:keyword**]**

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

sortcolumn **=** params**[**:sortcolumn**].**blank? **?** JobCategoryHelper**::**DEFAULT\_SORT\_COLUMN :

params**[**:sortcolumn**]**

sortdir **=** params**[**:sortdir**].**blank? **?** JobCategoryHelper**::**DEFAULT\_SORT\_DIR :

params**[**:sortdir**]**

sort **=** ApplicationHelper**::**Sort**.**new**(**sortcolumn**,** sortdir**)**

**if** keyword**.**blank?

@data **=** JobCategoryHelper**.**get\_all**(**pgnum**,** pgsize**,** sort**)**

**else**

@data **=** JobCategoryHelper**.**get\_filter\_by**(**keyword**,** pgnum**,** pgsize**,** sort**)**

**end**

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'list' **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# Display the JobCategory form.

# GET /jobcat/new

# GET /jobcat/new.json

**def** **new**

@jobcat **=** JobCategory**.**new

@form\_id **=** 'add-form'

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @jobcat **}**

**end**

**end**

# Create new JobCategory record.

# POST /jobcat/create

**def** **create**

o **=** JobCategory**.**new**(**:name **=>** params**[**:name**])**

**if** o**.**save

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Job Category was successfully added.' **}**

**else**

render :json **=>** JobCategoryHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

# Display the JobCategory form, with existing record to edit.

# GET /jobcat/edit/1

# GET /jobcat/edit/1.json

**def** **edit**

@jobcat **=** JobCategory**.**find**(**params**[**:id**])**

@form\_id **=** 'edit-form'

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @jobcat **}**

**end**

**end**

# Update JobCategory record.

# POST /jobcat/update/1

**def** **update**

o **=** JobCategory**.**find**(**params**[**:id**])**

**if** o**.**update\_attributes**(**:name **=>** params**[**:name**])**

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Job Category was successfully updated.' **}**

**else**

render :json **=>** JobCategoryHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

# Delete a list of JobCategory records.

# POST /jobcat/delete

**def** **destroy**

keyword **=** params**[**:keyword**].**blank? **?** '' : params**[**:keyword**]**

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

ids **=** params**[**:id**]**

JobCategory**.**delete\_all**(**:id **=>** ids**)**

itemscount **=** JobCategoryHelper**.**item\_message**(**keyword**,** pgnum**,** pgsize**)**

render :json **=>** **{** :success **=>** 1**,**

:itemscount **=>** itemscount**,**

:message **=>** %Q{**#{**ids**.**size**}** Job Categori(es) was successfully

deleted.} **}**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\admin\overtime\_chart\_controller.rb**

# This controller serves incoming requests to display Employee Overtime Chart.

**class** **Admin::**OvertimeChartController **<** Admin**::**AdminController

# Display the main page.

# GET /overtime/chart

**def** **index**

@month\_hash **=** month\_options

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:chart**]** **}**

**end**

**end**

# Get the overtime json data to be populated to the chart.

# GET /overtime/chart/data

**def** **data**

staff\_id **=** params**[**:staff\_id**].**blank? **?** '' : params**[**:staff\_id**]**

month **=** params**[**:month**].**blank? **?** '0' : params**[**:month**]**

year **=** params**[**:year**].**blank? **?** 0 : params**[**:year**].**to\_i

criteria **=** Attendance

title **=** 'Overtime'

yaxis **=** 'Duration (hours)'

**if** staff\_id**.**present?

criteria **=** criteria**.**where**(**:staff\_id **=>** staff\_id**)**

**end**

**if** year **!=** 0

criteria **=** criteria**.**where**(**'year(work\_date) = ?'**,** year**)**

title **=** "Overtime for Year **#{**year**}**"

**end**

**if** month **!=** '0'

criteria **=** criteria**.**where**(**'month(work\_date) in (?)'**,** month**)**

**end**

list **=** criteria**.**order**(**:work\_date**).**all

months **=** I18n**.**t**(**'date.month\_names'**)**

b **=** Array**.**new**(**12**)** **{** **|**x**|** 0 **}**

categories **=** Array**.**new**(**12**)** **{** **|**x**|** months**[**x **+** 1**][**0**..**2**]** **}**

list**.**each **do** **|**o**|**

to **=** ApplicationHelper**.**localtime**(**o**.**time\_out**)**

v **=** Time**.**new**(**to**.**year**,** to**.**month**,** to**.**day**,** 18**,** 0**,** 0**,** '+08:00'**)**

x **=** **(**to **-** v**)** **/** 3600.0

m **=** o**.**work\_date**.**month

**if** x **>** 0

b**[**m **-** 1**]** **+=** x

**end**

**end**

c **=** b**.**collect **{** **|**x**|** x**.**round**(**2**)** **}**

@data **=** **{** :data **=>** c**,** :categories **=>** categories**,** :title **=>** title**,** :yaxis **=>** yaxis **}**

render :json **=>** @data

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\admin\overtime\_rate\_controller.rb**

# This controller serves incoming requests to display out the OvertimeRate records.

**class** **Admin::**OvertimeRateController **<** Admin**::**AdminController

# GET /overtimerate

# GET /overtimerate.json

**def** **index**

@data **=** OvertimeRateHelper**.**get\_all

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:list**]** **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# List records by filtering.

# GET /overtimerate/list

# GET /overtimerate/list.json

**def** **list**

year **=** params**[**:year**].**blank? **?** 0 : params**[**:year**].**to\_i

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

sortcolumn **=** params**[**:sortcolumn**].**blank? **?** OvertimeRateHelper**::**DEFAULT\_SORT\_COLUMN :

params**[**:sortcolumn**]**

sortdir **=** params**[**:sortdir**].**blank? **?** OvertimeRateHelper**::**DEFAULT\_SORT\_DIR :

params**[**:sortdir**]**

sort **=** ApplicationHelper**::**Sort**.**new**(**sortcolumn**,** sortdir**)**

filters **=** **{** :year **=>** year **}**

**if** year **==** 0

@data **=** OvertimeRateHelper**.**get\_all**(**pgnum**,** pgsize**,** sort**)**

**else**

@data **=** OvertimeRateHelper**.**get\_filter\_by**(**filters**,** pgnum**,** pgsize**,** sort**)**

**end**

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'list' **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# Display the OvertimeRate form.

# GET /overtimerate/new

# GET /overtimerate/new.json

**def** **new**

@rate **=** OvertimeRate**.**new

@form\_id **=** 'add-form'

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @rate **}**

**end**

**end**

# Create new OvertimeRate record.

# POST /overtimerate/create

**def** **create**

o **=** OvertimeRate**.**new**(**:duration **=>** params**[**:duration**],** :year **=>** params**[**:year**],**

:pay\_rate **=>** params**[**:pay\_rate**])**

**if** o**.**save

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Overtime Rate successfully added.' **}**

**else**

render :json **=>** OvertimeRateHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

# Display the OvertimeRate form, with existing record to edit.

# GET /overtimerate/edit/1

# GET /overtimerate/edit/1.json

**def** **edit**

@rate **=** OvertimeRate**.**find**(**params**[**:id**])**

@form\_id **=** 'edit-form'

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @rate **}**

**end**

**end**

# Update OvertimeRate record.

# POST /overtimerate/update

**def** **update**

o **=** OvertimeRate**.**find**(**params**[**:id**])**

**if** o**.**update\_attributes**(**:duration **=>** params**[**:duration**],** :year **=>** params**[**:year**],**

:pay\_rate **=>** params**[**:pay\_rate**])**

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Pay Rate was successfully updated.' **}**

**else**

render :json **=>** OvertimeRateHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

# Delete a list of OvertimeRate records.

# POST /overtimerate/delete

**def** **destroy**

year **=** params**[**:year**].**blank? **?** 0 : params**[**:year**].**to\_i

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

ids **=** params**[**:id**]**

OvertimeRate**.**delete\_all**(**:id **=>** ids**)**

filters **=** **{** :year **=>** year **}**

**if** year **==** 0

itemscount **=** OvertimeRateHelper**.**item\_message**(nil,** pgnum**,** pgsize**)**

**else**

itemscount **=** OvertimeRateHelper**.**item\_message**(**filters**,** pgnum**,** pgsize**)**

**end**

render :json **=>** **{** :success **=>** 1**,**

:itemscount **=>** itemscount**,**

:message **=>** %Q{**#{**ids**.**size**}** Overtime rate(s) was successfully

deleted.} **}**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\admin\pay\_rate\_controller.rb**

require 'securerandom'

# This controller serves incoming requests to display out the PayRate records.

**class** **Admin::**PayRateController **<** Admin**::**AdminController

# GET /payrate

# GET /payrate.json

**def** **index**

@data **=** PayRateHelper**.**get\_all

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:list**]** **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# List records by filtering.

# GET /payrate/list

# GET /payrate/list.json

**def** **list**

staff\_id **=** params**[**:staff\_id**].**blank? **?** '' : params**[**:staff\_id**]**

month **=** params**[**:month**].**blank? **?** 0 : params**[**:month**].**to\_i

year **=** params**[**:year**].**blank? **?** 0 : params**[**:year**].**to\_i

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

sortcolumn **=** params**[**:sortcolumn**].**blank? **?** PayRateHelper**::**DEFAULT\_SORT\_COLUMN :

params**[**:sortcolumn**]**

sortdir **=** params**[**:sortdir**].**blank? **?** PayRateHelper**::**DEFAULT\_SORT\_DIR : params**[**:sortdir**]**

sort **=** ApplicationHelper**::**Sort**.**new**(**sortcolumn**,** sortdir**)**

filters **=** **{** :staff\_id **=>** staff\_id**,**

:month **=>** month**,**

:year **=>** year **}**

**if** staff\_id**.**blank? **&&** month **==** 0 **&&** year **==** 0

@data **=** PayRateHelper**.**get\_all**(**pgnum**,** pgsize**,** sort**)**

**else**

@data **=** PayRateHelper**.**get\_filter\_by**(**filters**,** pgnum**,** pgsize**,** sort**)**

**end**

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'list' **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# Display the PayRate form.

# GET /payrate/new

# GET /payrate/new.json

**def** **new**

@payrate **=** PayRate**.**new

@form\_id **=** 'add-form'

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @payrate **}**

**end**

**end**

# Create new PayRate record.

# POST /payrate/create

**def** **create**

o **=** PayRate**.**new**(**:id **=>** SecureRandom**.**uuid**,** :staff\_id **=>** params**[**:staff\_id**],**

:month **=>** params**[**:month**],** :year **=>** params**[**:year**],**

:hourly\_pay\_rate **=>** params**[**:pay\_rate**])**

**if** o**.**save

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Pay Rate successfully added.' **}**

**else**

render :json **=>** PayRateHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

# Display the PayRate form, with existing record to edit.

# GET /payrate/edit/1

# GET /payrate/edit/1.json

**def** **edit**

@payrate **=** PayRate**.**find**(**params**[**:id**])**

@form\_id **=** 'edit-form'

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @payrate **}**

**end**

**end**

# Update PayRate record.

# POST /payrate/update

**def** **update**

o **=** PayRate**.**find**(**params**[**:id**])**

**if** o**.**update\_attributes**(**:staff\_id **=>** params**[**:staff\_id**],** :month **=>** params**[**:month**],**

:year **=>** params**[**:year**],** :hourly\_pay\_rate **=>** params**[**:pay\_rate**])**

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Pay Rate was successfully updated.' **}**

**else**

render :json **=>** PayRateHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

# Delete a list of PayRate records.

# POST /payrate/delete

**def** **destroy**

staff\_id **=** params**[**:staff\_id**].**blank? **?** '' : params**[**:staff\_id**]**

month **=** params**[**:month**].**blank? **?** 0 : params**[**:month**].**to\_i

year **=** params**[**:year**].**blank? **?** 0 : params**[**:year**].**to\_i

status **=** params**[**:status**].**blank? **?** 0 : params**[**:status**].**to\_i

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

ids **=** params**[**:id**]**

PayRate**.**delete\_all**(**:id **=>** ids**)**

filters **=** **{** :staff\_id **=>** staff\_id**,**

:month **=>** month**,**

:year **=>** year **}**

**if** staff\_id**.**blank? **&&** month **==** 0 **&&** year **==** 0

itemscount **=** PayRateHelper**.**item\_message**(nil,** pgnum**,** pgsize**)**

**else**

itemscount **=** PayRateHelper**.**item\_message**(**filters**,** pgnum**,** pgsize**)**

**end**

render :json **=>** **{** :success **=>** 1**,**

:itemscount **=>** itemscount**,**

:message **=>** "**#{**ids**.**size**}** pay rate(s) was successfully deleted." **}**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\admin\payslip\_controller.rb**

# This controller serves incoming requests to display out the Employee records for pay

# slip generation.

**class** **Admin::**PayslipController **<** Admin**::**AdminController

# List all records.

# GET /payslip

# GET /payslip.json

**def** **index**

@data **=** EmployeeHelper**.**get\_all

@employmentstatus **=** EmploymentStatus**.**order**(**:name**).**all

@designation **=** Designation**.**order**(**:title**).**all

@dept **=** Department**.**order**(**:name**).**all

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:list**]** **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# List records by filtering.

# GET /payslip/list

# GET /payslip/list.json

**def** **list**

employee **=** params**[**:employee**].**blank? **?** '' : params**[**:employee**]**

staff\_id **=** params**[**:staff\_id**].**blank? **?** '' : params**[**:staff\_id**]**

employment\_status **=** params**[**:employment\_status**].**blank? **?** 0 :

params**[**:employment\_status**].**to\_i

designation **=** params**[**:designation**].**blank? **?** 0 : params**[**:designation**].**to\_i

dept **=** params**[**:dept**].**blank? **?** 0 : params**[**:dept**].**to\_i

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

sortcolumn **=** params**[**:sortcolumn**].**blank? **?** EmployeeHelper**::**DEFAULT\_SORT\_COLUMN :

params**[**:sortcolumn**]**

sortdir **=** params**[**:sortdir**].**blank? **?** EmployeeHelper**::**DEFAULT\_SORT\_DIR : params**[**:sortdir**]**

sort **=** ApplicationHelper**::**Sort**.**new**(**sortcolumn**,** sortdir**)**

filters **=** **{** :employee **=>** employee**,**

:staff\_id **=>** staff\_id**,**

:employment\_status **=>** employment\_status**,**

:designation **=>** designation**,**

:dept **=>** dept **}**

**if** employee**.**blank? **&&** staff\_id**.**blank? **&&** employment\_status **==** 0 **&&** designation **==** 0 **&&**

dept **==** 0

@data **=** EmployeeHelper**.**get\_all**(**pgnum**,** pgsize**,** sort**)**

**else**

@data **=** EmployeeHelper**.**get\_filter\_by**(**filters**,** pgnum**,** pgsize**,** sort**)**

**end**

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'list' **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# Display the pay slip information.

# GET /payslip/slip/1/1/2012

# GET /payslip/slip/1/1/2012.json

**def** **payslip**

@employee **=** Employee**.**find**(**params**[**:id**])**

@employee\_salary **=** @employee**.**employee\_salary

\_month **=** params**[**:month**].**to\_i

month **=** month\_name**(**params**[**:month**].**to\_i**)**

year **=** params**[**:year**].**blank? **?** Time**.**now**.**year : params**[**:year**].**to\_i

@period **=** "**#{**month**}**-**#{**year**}**"

**if** @employee\_salary**.**blank?

@total\_earnings **=** 0

@total\_deduct **=** 0

@nett\_salary **=** 0

@employee\_salary **=** EmployeeSalary**.**new

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render 'payslip\_monthly' **}**

fmt**.**json **{** render :json **=>** **[**@employee**,** @total\_earnings**,** @total\_deduct**,**

@nett\_salary**]** **}**

**end**

**else**

# checks whether the salary type is monthly

**if** @employee\_salary**.**pay\_type **==** 1

filters **=** **{** :year **=>** year**,** :month **=>** \_month**,** :staff\_id **=>** @employee**.**staff\_id **}**

@total\_overtime **=** PayslipHelper**.**total\_overtime**(**filters**)**

@total\_overtime\_earnings **=** PayslipHelper**.**total\_overtime\_earnings**(**filters**,**

@total\_overtime**)**

@adjustment **=** SalaryAdjustmentHelper**.**get\_salary\_adjustment**(**filters**)**

@total\_earnings **=** PayslipHelper**.**total\_earnings**(**@employee\_salary**,** @adjustment**,**

@total\_overtime\_earnings**)**

@total\_deduct **=** PayslipHelper**.**total\_deductions**(**@employee\_salary**)**

@nett\_salary **=** PayslipHelper**.**nett\_salary**(**@total\_earnings**,** @total\_deduct**)**

@basic\_pay **=** @employee\_salary**.**salary **+** @adjustment

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render 'payslip\_monthly' **}**

fmt**.**json **{** render :json **=>** **[**@employee**,** @total\_earnings**,** @total\_deduct**,**

@nett\_salary**,** @total\_overtime**,**

@total\_overtime\_earnings**,** @adjustment**,** @basic\_pay**]** **}**

**end**

# hourly type

**else**

filters **=** **{** :year **=>** year**,** :month **=>** \_month**,** :staff\_id **=>** @employee**.**staff\_id **}**

@total\_earnings**,** @total\_hours**,** @hourly\_pay\_rate **=**

PayslipHelper**.**total\_earnings\_hourly**(**@employee\_salary**,** filters**)**

@total\_deduct **=** PayslipHelper**.**total\_deductions**(**@employee\_salary**)**

@nett\_salary **=** PayslipHelper**.**nett\_salary\_hourly**(**@total\_earnings**,** @total\_deduct**)**

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render 'payslip\_hourly' **}**

fmt**.**json **{** render :json **=>** **[**@employee**,** @total\_earnings**,** @total\_deduct**,**

@nett\_salary**,** @total\_hours**,** @hourly\_pay\_rate**]** **}**

**end**

**end**

**end**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\admin\salary\_adjustment\_controller.rb**

require 'securerandom'

# This controller serves incoming requests to display out the SalaryAdjustment records.

**class** **Admin::**SalaryAdjustmentController **<** Admin**::**AdminController

# GET /salaryadj

# GET /salaryadj.json

**def** **index**

@data **=** SalaryAdjustmentHelper**.**get\_all

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:list**]** **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# List records by filtering.

# GET /salaryadj/list

# GET /salaryadj/list.json

**def** **list**

staff\_id **=** params**[**:staff\_id**].**blank? **?** '' : params**[**:staff\_id**]**

month **=** params**[**:month**].**blank? **?** 0 : params**[**:month**].**to\_i

year **=** params**[**:year**].**blank? **?** 0 : params**[**:year**].**to\_i

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

sortcolumn **=** params**[**:sortcolumn**].**blank? **?** SalaryAdjustmentHelper**::**DEFAULT\_SORT\_COLUMN :

params**[**:sortcolumn**]**

sortdir **=** params**[**:sortdir**].**blank? **?** SalaryAdjustmentHelper**::**DEFAULT\_SORT\_DIR :

params**[**:sortdir**]**

sort **=** ApplicationHelper**::**Sort**.**new**(**sortcolumn**,** sortdir**)**

filters **=** **{** :staff\_id **=>** staff\_id**,**

:month **=>** month**,**

:year **=>** year **}**

**if** staff\_id**.**blank? **&&** month **==** 0 **&&** year **==** 0

@data **=** SalaryAdjustmentHelper**.**get\_all**(**pgnum**,** pgsize**,** sort**)**

**else**

@data **=** SalaryAdjustmentHelper**.**get\_filter\_by**(**filters**,** pgnum**,** pgsize**,** sort**)**

**end**

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'list' **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# Display the SalaryAdjustment form.

# GET /salaryadj/new

# GET /salaryadj/new.json

**def** **new**

@adj **=** SalaryAdjustment**.**new

@form\_id **=** 'add-form'

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @adj **}**

**end**

**end**

# Create new SalaryAdjustment record.

# POST /salaryadj/create

**def** **create**

staff\_id **=** params**[**:staff\_id**]**

o **=** SalaryAdjustment**.**new**(**:id **=>** SecureRandom**.**uuid**,** :staff\_id **=>** params**[**:staff\_id**],**

:inc **=>** params**[**:inc**],** :month **=>** params**[**:month**],**

:year **=>** params**[**:year**])**

**if** o**.**save

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Salary Adjustment successfully added.' **}**

**else**

render :json **=>** SalaryAdjustmentHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

# Display the SalaryAdjustment form, with existing record to edit.

# GET /salaryadj/edit/1

# GET /salaryadj/edit/1.json

**def** **edit**

@adj **=** SalaryAdjustment**.**find**(**params**[**:id**])**

@form\_id **=** 'edit-form'

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @adj **}**

**end**

**end**

# Update SalaryAdjustment record.

# POST /salaryadj/update

**def** **update**

o **=** SalaryAdjustment**.**find**(**params**[**:id**])**

**if** o**.**update\_attributes**(**:staff\_id **=>** params**[**:staff\_id**],** :inc **=>** params**[**:inc**],**

:month **=>** params**[**:month**],** :year **=>** params**[**:year**])**

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Pay Rate was successfully updated.' **}**

**else**

render :json **=>** PayRateHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

# Delete a list of SalaryAdjustment records.

# POST /salaryadj/delete

**def** **destroy**

staff\_id **=** params**[**:staff\_id**].**blank? **?** '' : params**[**:staff\_id**]**

month **=** params**[**:month**].**blank? **?** 0 : params**[**:month**].**to\_i

year **=** params**[**:year**].**blank? **?** 0 : params**[**:year**].**to\_i

status **=** params**[**:status**].**blank? **?** 0 : params**[**:status**].**to\_i

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

ids **=** params**[**:id**]**

SalaryAdjustment**.**delete\_all**(**:id **=>** ids**)**

filters **=** **{** :staff\_id **=>** staff\_id**,**

:month **=>** month**,**

:year **=>** year **}**

**if** staff\_id**.**blank? **&&** month **==** 0 **&&** year **==** 0

itemscount **=** SalaryAdjustmentHelper**.**item\_message**(nil,** pgnum**,** pgsize**)**

**else**

itemscount **=** SalaryAdjustmentHelper**.**item\_message**(**filters**,** pgnum**,** pgsize**)**

**end**

render :json **=>** **{** :success **=>** 1**,**

:itemscount **=>** itemscount**,**

:message **=>** %Q{**#{**ids**.**size**}** salary adjustment(s) was successfully

deleted.} **}**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\admin\total\_work\_hours\_chart\_controller.rb**

# This controller serves incoming requests to display Employee Total Hours Worked Chart.

**class** **Admin::**TotalWorkHoursChartController **<** Admin**::**AdminController

# Display the main page.

# GET /workhours/chart

**def** **index**

@month\_hash **=** month\_options

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:chart**]** **}**

**end**

**end**

# Get the total hours worked json data to be populated to the chart.

# GET /workhours/chart/data

**def** **data**

staff\_id **=** params**[**:staff\_id**].**blank? **?** '' : params**[**:staff\_id**]**

month **=** params**[**:month**].**blank? **?** '0' : params**[**:month**]**

year **=** params**[**:year**].**blank? **?** 0 : params**[**:year**].**to\_i

title **=** 'Total Hours Worked'

yaxis **=** 'Duration (hours)'

months **=** I18n**.**t**(**'date.month\_names'**)**

b **=** Array**.**new**(**12**)** **{** **|**x**|** 0 **}**

categories **=** Array**.**new**(**12**)** **{** **|**x**|** months**[**x **+** 1**][**0**..**2**]** **}**

**if** staff\_id**.**present?

liststaff **=** **[**staff\_id**]**

**else**

list **=** PayRate**.**select**(**'distinct(staff\_id)'**).**all

liststaff **=** list**.**collect **{** **|**x**|** x**.**staff\_id **}**

**end**

**if** year **!=** 0

listyear **=** **[**year**]**

title **=** "Total Hours Worked for **#{**year**}**"

**else**

list **=** PayRate**.**select**(**'distinct(year)'**).**all

listyear **=** list**.**collect **{** **|**x**|** x**.**year **}**

**end**

**if** month **!=** '0'

listmonth **=** month**.**collect **{** **|**x**|** x**.**to\_i **}**

**else**

list **=** PayRate**.**select**(**'distinct(month)'**).**all

listmonth **=** list**.**collect **{** **|**x**|** x**.**month **}**

**end**

listyear**.**each **do** **|**y**|**

listmonth**.**each **do** **|**m**|**

liststaff**.**each **do** **|**s**|**

filters **=** **{** :year **=>** y**,** :month **=>** m**,** :staff\_id **=>** s **}**

total\_hours **=** AttendanceHelper**.**get\_total\_hours**(**filters**)**

v **=** total\_hours

b**[**m **-** 1**]** **+=** v

**end**

**end**

**end**

c **=** b**.**collect **{** **|**x**|** x**.**round**(**2**)** **}**

@data **=** **{** :data **=>** c**,** :categories **=>** categories**,** :title **=>** title**,** :yaxis **=>** yaxis **}**

render :json **=>** @data

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\admin\user\_controller.rb**

require 'securerandom'

# This controller serves incoming requests to display out the User records.

**class** **Admin::**UserController **<** Admin**::**AdminController

# List all records.

# GET /user

# GET /user.json

**def** **index**

@data **=** UserHelper**.**get\_all

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:list**]** **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# List records by filtering.

# GET /user/list

# GET /user/list.json

**def** **list**

username **=** params**[**:username**].**blank? **?** '' : params**[**:username**]**

role **=** params**[**:role**].**blank? **?** 0 : params**[**:role**].**to\_i

employee **=** params**[**:employee**].**blank? **?** '' : params**[**:employee**]**

status **=** params**[**:status**].**blank? **?** 0 : params**[**:status**].**to\_i

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

sortcolumn **=** params**[**:sortcolumn**].**blank? **?** UserHelper**::**DEFAULT\_SORT\_COLUMN :

params**[**:sortcolumn**]**

sortdir **=** params**[**:sortdir**].**blank? **?** UserHelper**::**DEFAULT\_SORT\_DIR : params**[**:sortdir**]**

sort **=** ApplicationHelper**::**Sort**.**new**(**sortcolumn**,** sortdir**)**

filters **=** **{** :username **=>** username**,**

:role **=>** role**,**

:employee **=>** employee**,**

:status **=>** status **}**

**if** username**.**blank? **&&** role **==** 0 **&&** employee**.**blank? **&&** status **==** 0

@data **=** UserHelper**.**get\_all**(**pgnum**,** pgsize**,** sort**)**

**else**

@data **=** UserHelper**.**get\_filter\_by**(**filters**,** pgnum**,** pgsize**,** sort**)**

**end**

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'list' **}**

fmt**.**json **{** render :json **=>** @data **}**

**end**

**end**

# Display the User form.

# GET /user/new

# GET /user/new.json

**def** **new**

@user **=** User**.**new

@form\_id **=** 'add-form'

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @user **}**

**end**

**end**

# Create new User record.

# POST /user/create

**def** **create**

status **=** params**[**:status**]**

status **=** status **==** '1' **?** **true** : **false**

o **=** User**.**new**(**:id **=>** SecureRandom**.**uuid**,** :role **=>** params**[**:role**],**

:username **=>** params**[**:username**],** :status **=>** status**,**

:pwd **=>** params**[**:pwd**],** :pwd\_confirmation **=>** params**[**:pwdconfirm**])**

**if** o**.**save

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'User was successfully added.' **}**

**else**

render :json **=>** UserHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

# Display the User form, with existing record to edit.

# GET /user/edit/1

# GET /user/edit/1.json

**def** **edit**

@user **=** User**.**find**(**params**[**:id**])**

@form\_id **=** 'edit-form'

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :partial **=>** 'form' **}**

fmt**.**json **{** render :json **=>** @user **}**

**end**

**end**

# Update User record.

# POST /user/update

**def** **update**

o **=** User**.**find**(**params**[**:id**])**

status **=** params**[**:status**]**

status **=** status **==** '1' **?** **true** : **false**

**if** o**.**update\_attributes**(**:role **=>** params**[**:role**],** :username **=>** params**[**:username**],**

:status **=>** status**,** :pwd **=>** params**[**:pwd**],**

:pwd\_confirmation **=>** params**[**:pwdconfirm**])**

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'User was successfully updated.' **}**

**else**

render :json **=>** UserHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

# Delete a list of User records.

# POST /user/delete

**def** **destroy**

username **=** params**[**:username**].**blank? **?** '' : params**[**:username**]**

role **=** params**[**:role**].**blank? **?** 0 : params**[**:role**].**to\_i

employee **=** params**[**:employee**].**blank? **?** '' : params**[**:employee**]**

status **=** params**[**:status**].**blank? **?** 0 : params**[**:status**].**to\_i

pgnum **=** params**[**:pgnum**].**blank? **?** 1 : params**[**:pgnum**].**to\_i

pgsize **=** params**[**:pgsize**].**blank? **?** 0 : params**[**:pgsize**].**to\_i

ids **=** params**[**:id**]**

User**.**delete\_all**(**:id **=>** ids**)**

filters **=** **{** :username **=>** username**,**

:role **=>** role**,**

:employee **=>** employee**,**

:status **=>** status **}**

**if** username**.**blank? **&&** role **==** 0 **&&** employee**.**blank? **&&** status **==** 0

itemscount **=** UserHelper**.**item\_message**(nil,** pgnum**,** pgsize**)**

**else**

itemscount **=** UserHelper**.**item\_message**(**filters**,** pgnum**,** pgsize**)**

**end**

render :json **=>** **{** :success **=>** 1**,**

:itemscount **=>** itemscount**,**

:message **=>** "**#{**ids**.**size**}** user(s) was successfully deleted." **}**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\user\contact\_controller.rb**

# This controller serves incoming requests to display out the EmployeeContact record.

**class** **User::**ContactController **<** User**::**UserController

# Display the contact.

# GET /contact

# GET /contact.json

**def** **index**

id **=** get\_employee\_id

@employee\_contact **=** EmployeeContactHelper**.**find**(**id**)**

respond\_to **do** **|**fmt**|**

fmt**.**html

fmt**.**json **{** render :json **=>** @employee\_contact **}**

**end**

**end**

# Update the contact.

# POST /contact/update

**def** **update**

id **=** get\_employee\_id

oc **=** EmployeeContactHelper**.**find**(**id**)**

oc\_new **=** **false**

**if** oc**.**id**.**blank?

o **=** Employee**.**find**(**id**)**

oc **=** EmployeeContactHelper**.**employee\_contact\_obj**(**o**,** params**)**

oc\_new **=** **true**

**end**

**if** oc\_new

**if** oc**.**save

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Contact Details was successfully updated.' **}**

**else**

render :json **=>** EmployeeContactHelper**.**get\_errors**(**oc**.**errors**)**

**end**

**else**

**if** EmployeeContactHelper**.**update\_obj**(**oc**,** params**)**

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Contact Details was successfully updated.' **}**

**else**

render :json **=>** EmployeeContactHelper**.**get\_errors**(**oc**.**errors**)**

**end**

**end**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\user\hourly\_payroll\_chart\_controller.rb**

# This controller serves incoming requests to display Hourly Payroll Chart.

**class** **User::**HourlyPayrollChartController **<** User**::**UserController

# Display the main page.

# GET /hourly/chart

**def** **index**

@month\_hash **=** month\_options

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:chart**]** **}**

**end**

**end**

# Get the hourly payroll json data to be populated to the chart.

# POST /hourly/chart/data

**def** **data**

month **=** params**[**:month**].**blank? **?** '0' : params**[**:month**]**

year **=** params**[**:year**].**blank? **?** 0 : params**[**:year**].**to\_i

staff\_id **=** get\_staff\_id

title **=** 'Hourly Payroll'

yaxis **=** 'Total Amount (RM)'

months **=** I18n**.**t**(**'date.month\_names'**)**

o **=** Array**.**new**(**12**)** **{** **|**x**|** **[**months**[**x **+** 1**],** 0**]** **}**

b **=** Array**.**new**(**12**)** **{** **|**x**|** 0 **}**

categories **=** Array**.**new**(**12**)** **{** **|**x**|** months**[**x **+** 1**][**0**..**2**]** **}**

**if** year **!=** 0

listyear **=** **[**year**]**

title **=** "Hourly Payroll for **#{**year**}**"

**else**

list **=** PayRate**.**select**(**'distinct(year)'**).**all

listyear **=** list**.**collect **{** **|**x**|** x**.**year **}**

**end**

**if** month **!=** '0'

listmonth **=** month**.**collect **{** **|**x**|** x**.**to\_i **}**

**else**

list **=** PayRate**.**select**(**'distinct(month)'**).**all

listmonth **=** list**.**collect **{** **|**x**|** x**.**month **}**

**end**

listyear**.**each **do** **|**y**|**

listmonth**.**each **do** **|**m**|**

filters **=** **{** :year **=>** y**,** :month **=>** m**,** :staff\_id **=>** staff\_id **}**

total\_hours **=** AttendanceHelper**.**get\_total\_hours**(**filters**)**

rate **=** PayRateHelper**.**get\_pay\_rate**(**filters**)**

v **=** total\_hours **\*** rate

o**[**m **-** 1**][**1**]** **=** v

b**[**m **-** 1**]** **+=** v

**end**

**end**

c **=** b**.**collect **{** **|**x**|** x**.**round**(**2**)** **}**

@data **=** **{** :pie **=>** o**,** :column **=>** **{** :data **=>** c**,**

:categories **=>** categories**,**

:yaxis **=>** yaxis **},**

:title **=>** title **}**

render :json **=>** @data

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\user\info\_controller.rb**

# This controller serves incoming requests to display out the Employee record.

**class** **User::**InfoController **<** User**::**UserController

# Display the information.

# GET /info

# GET /info.json

**def** **index**

id **=** get\_employee\_id

@employee **=** Employee**.**find**(**id**)**

@user **=** @employee**.**user

respond\_to **do** **|**fmt**|**

fmt**.**html

fmt**.**json **{** render :json **=>** @employee **}**

**end**

**end**

# Update the information.

# POST /info/update

**def** **update**

id **=** get\_employee\_id

o **=** Employee**.**find**(**id**)**

**if** EmployeeHelper**.**update\_info**(**o**,** params**)**

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Personal Details was successfully updated.' **}**

**else**

render :json **=>** EmployeeHelper**.**get\_errors**(**o**.**errors**)**

**end**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\user\job\_controller.rb**

# This controller serves incoming requests to display out the EmployeeJob record.

**class** **User::**JobController **<** User**::**UserController

# Display the information.

# GET /job

# GET /job.json

**def** **index**

id **=** get\_employee\_id

@employee\_job **=** EmployeeJobHelper**.**find**(**id**)**

respond\_to **do** **|**fmt**|**

fmt**.**html

fmt**.**json **{** render :json **=>** @employee\_job **}**

**end**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\user\overtime\_chart\_controller.rb**

# This controller serves incoming requests to display Employee Overtime Chart.

**class** **User::**OvertimeChartController **<** User**::**UserController

# Display the main page.

# GET /overtime/chart

**def** **index**

@month\_hash **=** month\_options

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:chart**]** **}**

**end**

**end**

# Get the overtime json data to be populated to the chart.

# GET /overtime/chart/data

**def** **data**

month **=** params**[**:month**].**blank? **?** '0' : params**[**:month**]**

year **=** params**[**:year**].**blank? **?** 0 : params**[**:year**].**to\_i

staff\_id **=** get\_staff\_id

criteria **=** Attendance

title **=** 'Overtime'

yaxis **=** 'Duration (hours)'

criteria **=** criteria**.**where**(**:staff\_id **=>** staff\_id**)**

**if** year **!=** 0

criteria **=** criteria**.**where**(**'year(work\_date) = ?'**,** year**)**

title **=** "Overtime for Year **#{**year**}**"

**end**

**if** month **!=** '0'

criteria **=** criteria**.**where**(**'month(work\_date) in (?)'**,** month**)**

**end**

list **=** criteria**.**order**(**:work\_date**).**all

months **=** I18n**.**t**(**'date.month\_names'**)**

b **=** Array**.**new**(**12**)** **{** **|**x**|** 0 **}**

categories **=** Array**.**new**(**12**)** **{** **|**x**|** months**[**x **+** 1**][**0**..**2**]** **}**

list**.**each **do** **|**o**|**

to **=** ApplicationHelper**.**localtime**(**o**.**time\_out**)**

v **=** Time**.**new**(**to**.**year**,** to**.**month**,** to**.**day**,** 18**,** 0**,** 0**,** '+08:00'**)**

x **=** **(**to **-** v**)** **/** 3600.0

m **=** o**.**work\_date**.**month

b**[**m **-** 1**]** **+=** x

**end**

c **=** b**.**collect **{** **|**x**|** x**.**round**(**2**)** **}**

@data **=** **{** :data **=>** c**,** :categories **=>** categories**,** :title **=>** title**,** :yaxis **=>** yaxis **}**

render :json **=>** @data

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\user\payslip\_controller.rb**

# This controller serves incoming requests to display out the Employee pay slip.

**class** **User::**PayslipController **<** User**::**UserController

# Display the main page.

# GET /payslip

**def** **index**

respond\_to **do** **|**fmt**|**

fmt**.**html

**end**

**end**

# Display the pay slip information.

# GET /payslip/slip

# GET /payslip/slip.json

**def** **payslip**

id **=** get\_employee\_id

@employee **=** Employee**.**find**(**id**)**

@employee\_salary **=** @employee**.**employee\_salary

\_month **=** params**[**:month**].**to\_i

month **=** month\_name**(**params**[**:month**].**to\_i**)**

year **=** params**[**:year**].**blank? **?** Time**.**now**.**year : params**[**:year**].**to\_i

@period **=** "**#{**month**}**-**#{**year**}**"

**if** @employee\_salary**.**blank?

@total\_earnings **=** 0

@total\_deduct **=** 0

@nett\_salary **=** 0

@employee\_salary **=** EmployeeSalary**.**new

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render 'admin/payslip/payslip\_monthly' **}**

fmt**.**json **{** render :json **=>** **[**@employee**,** @total\_earnings**,** @total\_deduct**,**

@nett\_salary**]** **}**

**end**

**else**

# checks whether the salary type is monthly

**if** @employee\_salary**.**pay\_type **==** 1

filters **=** **{** :year **=>** year**,** :month **=>** \_month**,** :staff\_id **=>** @employee**.**staff\_id **}**

@total\_overtime **=** PayslipHelper**.**total\_overtime**(**filters**)**

@total\_overtime\_earnings **=** PayslipHelper**.**total\_overtime\_earnings**(**filters**,**

@total\_overtime**)**

@adjustment **=** SalaryAdjustmentHelper**.**get\_salary\_adjustment**(**filters**)**

@total\_earnings **=** PayslipHelper**.**total\_earnings**(**@employee\_salary**,** @adjustment**,**

@total\_overtime\_earnings**)**

@total\_deduct **=** PayslipHelper**.**total\_deductions**(**@employee\_salary**)**

@nett\_salary **=** PayslipHelper**.**nett\_salary**(**@total\_earnings**,** @total\_deduct**)**

@basic\_pay **=** @employee\_salary**.**salary **+** @adjustment

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render 'admin/payslip/payslip\_monthly' **}**

fmt**.**json **{** render :json **=>** **[**@employee**,** @total\_earnings**,** @total\_deduct**,**

@nett\_salary**,** @total\_overtime**,**

@total\_overtime\_earnings**,** @adjustment**,** @basic\_pay**]** **}**

**end**

# hourly type

**else**

filters **=** **{** :year **=>** year**,** :month **=>** \_month**,** :staff\_id **=>** @employee**.**staff\_id **}**

@total\_earnings**,** @total\_hours**,** @hourly\_pay\_rate **=**

PayslipHelper**.**total\_earnings\_hourly**(**@employee\_salary**,** filters**)**

@total\_deduct **=** PayslipHelper**.**total\_deductions**(**@employee\_salary**)**

@nett\_salary **=** PayslipHelper**.**nett\_salary\_hourly**(**@total\_earnings**,** @total\_deduct**)**

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render 'admin/payslip/payslip\_hourly' **}**

fmt**.**json **{** render :json **=>** **[**@employee**,** @total\_earnings**,** @total\_deduct**,**

@nett\_salary**,** @total\_hours**,** @hourly\_pay\_rate**]** **}**

**end**

**end**

**end**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\user\qualification\_controller.rb**

# This controller serves incoming requests to display out the EmployeeQualification record.

**class** **User::**QualificationController **<** User**::**UserController

# Display the qualification.

# GET /qualification

# GET /qualification.json

**def** **index**

id **=** get\_employee\_id

@employee\_qualification **=** EmployeeQualificationHelper**.**find**(**id**)**

respond\_to **do** **|**fmt**|**

fmt**.**html

fmt**.**json **{** render :json **=>** @employee\_qualification **}**

**end**

**end**

# Update the qualification.

# POST /qualification/update

**def** **update**

id **=** get\_employee\_id

oq **=** EmployeeQualificationHelper**.**find**(**id**)**

oq\_new **=** **false**

**if** oq**.**id**.**blank?

o **=** Employee**.**find**(**id**)**

oq **=** EmployeeQualificationHelper**.**employee\_qualification\_obj**(**o**,** params**)**

oq\_new **=** **true**

**end**

**if** oq\_new

**if** oq**.**save

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Qualifications was successfully updated.' **}**

**else**

render :json **=>** EmployeeQualificationHelper**.**get\_errors**(**oq**.**errors**)**

**end**

**else**

**if** EmployeeQualificationHelper**.**update\_obj**(**oq**,** params**)**

render :json **=>** **{** :success **=>** 1**,**

:message **=>** 'Qualifications was successfully updated.' **}**

**else**

render :json **=>** EmployeeQualificationHelper**.**get\_errors**(**oq**.**errors**)**

**end**

**end**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\user\salary\_controller.rb**

# This controller serves incoming requests to display out the EmployeeSalary record.

**class** **User::**SalaryController **<** User**::**UserController

# Display the information.

# GET /salary

# GET /salary.json

**def** **index**

id **=** get\_employee\_id

@employee **=** Employee**.**find**(**id**)**

@employee\_salary **=** EmployeeSalaryHelper**.**find**(**id**)**

@adjustment **=** SalaryAdjustmentHelper**.**get\_salary\_adjustment**(**

:staff\_id **=>** @employee**.**staff\_id**,** :year **=>** Time**.**now**.**year**)**

@basic\_pay **=** @employee\_salary**.**salary **+** @adjustment

respond\_to **do** **|**fmt**|**

fmt**.**html

fmt**.**json **{** render :json **=>** **[**@employee\_salary**,** @adjustment**]** **}**

**end**

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\user\total\_work\_hours\_chart\_controller.rb**

# This controller serves incoming requests to display Employee Total Hours Worked Chart.

**class** **User::**TotalWorkHoursChartController **<** User**::**UserController

# Display the main page.

# GET /workhours/chart

**def** **index**

@month\_hash **=** month\_options

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:chart**]** **}**

**end**

**end**

# Get the total hours worked json data to be populated to the chart.

# GET /workhours/chart/data

**def** **data**

month **=** params**[**:month**].**blank? **?** '0' : params**[**:month**]**

year **=** params**[**:year**].**blank? **?** 0 : params**[**:year**].**to\_i

staff\_id **=** get\_staff\_id

title **=** 'Total Hours Worked'

yaxis **=** 'Duration (hours)'

months **=** I18n**.**t**(**'date.month\_names'**)**

b **=** Array**.**new**(**12**)** **{** **|**x**|** 0 **}**

categories **=** Array**.**new**(**12**)** **{** **|**x**|** months**[**x **+** 1**][**0**..**2**]** **}**

liststaff **=** **[**staff\_id**]**

**if** year **!=** 0

listyear **=** **[**year**]**

title **=** "Total Hours Worked for **#{**year**}**"

**else**

list **=** PayRate**.**select**(**'distinct(year)'**).**all

listyear **=** list**.**collect **{** **|**x**|** x**.**year **}**

**end**

**if** month **!=** '0'

listmonth **=** month**.**collect **{** **|**x**|** x**.**to\_i **}**

**else**

list **=** PayRate**.**select**(**'distinct(month)'**).**all

listmonth **=** list**.**collect **{** **|**x**|** x**.**month **}**

**end**

listyear**.**each **do** **|**y**|**

listmonth**.**each **do** **|**m**|**

liststaff**.**each **do** **|**s**|**

filters **=** **{** :year **=>** y**,** :month **=>** m**,** :staff\_id **=>** s **}**

total\_hours **=** AttendanceHelper**.**get\_total\_hours**(**filters**)**

v **=** total\_hours

b**[**m **-** 1**]** **+=** v

**end**

**end**

**end**

c **=** b**.**collect **{** **|**x**|** x**.**round**(**2**)** **}**

@data **=** **{** :data **=>** c**,** :categories **=>** categories**,** :title **=>** title**,** :yaxis **=>** yaxis **}**

render :json **=>** @data

**end**

**end**

**E:\workspace\payroll\_rails\app\controllers\user\user\_controller.rb**

# User base controller.

# All User controllers must inherit this controller.

# User controller serves incoming requests from an authenticated Normal User.

**class** **User::**UserController **<** ApplicationController

layout **false**

# checks whether a user is authenticated before serving any request

before\_filter :authenticate\_normal\_user

# Display the user page.

# GET /user/index

**def** **index**

id **=** get\_employee\_id

@employee\_salary **=** EmployeeSalaryHelper**.**find**(**id**)**

@pay\_type **=** @employee\_salary**.**pay\_type

respond\_to **do** **|**fmt**|**

fmt**.**html **{** render :layout **=>** LAYOUT**[**:user**]** **}**

**end**

**end**

protected

# Returns the employee\_id.

**def** **normal\_user**

**return** unless session**[**:employee\_id**]**

@normal\_user **||=** session**[**:employee\_id**]**

**end**

# Checks whether a normal user is authenticated.

**def** **authenticate\_normal\_user**

**unless** authenticate

**return** **false**

**end**

logged\_in\_normal\_user? **?** **true** : access\_denied

**end**

# Checks whether a normal user is logged in.

**def** **logged\_in\_normal\_user?**

normal\_user**.**present?

**end**

# Returns the employee\_id.

**def** **get\_employee\_id**

session**[**:employee\_id**]**

**end**

# Returns the staff\_id.

**def** **get\_staff\_id**

session**[**:staff\_id**]**

**end**

helper\_method :normal\_user

helper\_method :logged\_in\_normal\_user

**end**

**E:\workspace\payroll\_rails\app\helpers\application\_helper.rb**

**module** **ApplicationHelper**

# Pager object for paging purpose

**class** **Pager**

attr\_accessor :total**,** :pagenum

attr\_reader :pagesize

@@default\_page\_size **=** 50

**def** **initialize(**total**,** pagenum**,** pagesize**)**

@total **=** total

@pagenum **=** pagenum

set\_pagesize**(**pagesize**)**

**end**

**def** **pagesize=(**pagesize**)**

set\_pagesize**(**pagesize**)**

**end**

# Get the paging message.

**def** **item\_message**

Utils**.**item\_message**(**total**,** pagenum**,** pagesize**)**

**end**

# Get the lower bound of the page.

**def** **lower\_bound**

**(**pagenum **-** 1**)** **\*** pagesize

**end**

# Get the upper bound of the page.

**def** **upper\_bound**

upperbound **=** pagenum **\*** pagesize

**if** total **<** upperbound

upperbound **=** total

**end**

upperbound

**end**

# Checks whether the pager has next page.

**def** **has\_next?**

total **>** upper\_bound **?** **true** : **false**

**end**

# Checks whether the pager has previous page.

**def** **has\_prev?**

lower\_bound **>** 0 **?** **true** : **false**

**end**

# Get the total pages.

**def** **total\_pages**

**(**Float**(**total**)** **/** pagesize**).**ceil

**end**

# Get the default page size.

**def** **self.**default\_page\_size

@@default\_page\_size

**end**

private

**def** **set\_pagesize(**pagesize**)**

**if** **(**total **<** pagesize **||** pagesize **<** 1**)** **&&** total **>** 0

@pagesize **=** total

**else**

@pagesize **=** pagesize

**end**

**if** total\_pages **<** @pagenum

@pagenum **=** total\_pages

**end**

**if** @pagenum **<** 1

@pagenum **=** 1

**end**

**end**

**end**

**module** **Utils**

require 'time'

# Get the paging message.

**def** **self.**item\_message**(**total**,** pagenum**,** pagesize**)**

x **=** **(**pagenum **-** 1**)** **\*** pagesize **+** 1

y **=** pagenum **\*** pagesize

**if** total **<** y

y **=** total

**end**

**if** total **<** 1

**return** ""

**end**

"**#{**x**}** to **#{**y**}** of **#{**total**}**"

**end**

# Checks for numeric.

**def** **self.**numeric?**(**val**)**

**true** if Float**(**val**)** **rescue** **false**

**end**

**end**

**class** **Sort**

attr\_accessor :column**,** :direction

**def** **initialize(**column**,** dir **=** 'ASC'**)**

@column **=** column

@direction **=** dir

**end**

**def** **to\_s**

"**#{**@column**}** **#{**@direction**}**"

**end**

**end**

# Get month name for the specified month number,

# e.g return January for 1

**def** **self.**month\_name**(**i**)**

Date**::**MONTHNAMES**[**i**]**

**end**

**def** **self.**localtime**(**t**)**

t**.**in\_time\_zone**(**'Kuala Lumpur'**)**

**end**

**def** **self.**date\_fmt

'%d-%m-%Y'

**end**

**end**

**E:\workspace\payroll\_rails\app\helpers\attendance\_helper.rb**

**module** **AttendanceHelper**

DEFAULT\_SORT\_COLUMN **=** 'work\_date'

DEFAULT\_SORT\_DIR **=** 'ASC'

# Get all attendance records.

**def** **self.**get\_all**(**pagenum **=** 1**,** pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

total **=** Attendance**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

order **=** sort**.**to\_s

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

**if** sort**.**column **==** 'e.first\_name'

criteria **=** get\_join**({},** sort**)**

**else**

criteria **=** Attendance

**end**

list **=** criteria**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get filtered attendance records.

**def** **self.**get\_filter\_by**(**filters**,** pagenum **=** 1**,**

pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

criteria**,** order **=** get\_filter\_criteria**(**filters**,** sort**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** criteria**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get the total hours worked

**def** **self.**get\_total\_hours**(**filters**)**

criteria **=** Attendance**.**where**(**'year(work\_date) = ? and month(work\_date) = ?'**,**

filters**[**:year**],** filters**[**:month**])**

**if** filters**[**:staff\_id**].**present?

criteria **=** criteria**.**where**(**'staff\_id = ?'**,** filters**[**:staff\_id**])**

**end**

list **=** criteria**.**all

total\_hours **=** 0

list**.**each **do** **|**o**|**

to **=** ApplicationHelper**.**localtime**(**o**.**time\_out**.**localtime**)**

ti **=** ApplicationHelper**.**localtime**(**o**.**time\_in**.**localtime**)**

total\_hours **+=** **(**to **-** ti**)** **/** 3600.0

**end**

total\_hours

**end**

private

**def** **self.**get\_filter\_criteria**(**filters**,** sort **=** **nil)**

employee\_keyword **=** "%**#{**filters**[**:employee**]}**%"

order **=** sort**.**present? **?** sort**.**to\_s : **nil**

**if** filters**[**:employee**].**present? **||** sort**.**present?

criteria **=** get\_join**(**filters**,** sort**)**

**end**

criteria **=** Attendance if criteria**.**blank?

**if** filters**[**:work\_date**].**present?

criteria **=** criteria**.**where**(**'work\_date = ?'**,** filters**[**:work\_date**])**

**end**

**if** filters**[**:employee**].**present?

criteria **=** criteria**.**where**(**

'e.first\_name like ? or e.middle\_name like ? or e.last\_name like ?'**,**

employee\_keyword**,** employee\_keyword**,** employee\_keyword**)**

**end**

**return** criteria**,** order

**end**

**def** **self.**get\_join**(**filters**,** sort **=** **nil)**

joinhash **=** **{}**

**if** filters**.**any?

**if** filters**[**:employee**].**present?

q **=** Attendance**.**joins**(**'inner join employee e on attendance.staff\_id = e.staff\_id'**)**

joinhash**[**:employee**]** **=** **true**

**end**

**end**

**if** sort**.**present?

**if** sort**.**column **==** 'e.first\_name'

**if** **!**joinhash**.**has\_key?**(**:employee**)**

q **=** Attendance**.**joins**(**

'left outer join employee e on attendance.staff\_id = e.staff\_id'**)**

**end**

**end**

**end**

q

**end**

**end**

**E:\workspace\payroll\_rails\app\helpers\department\_helper.rb**

**module** **DepartmentHelper**

DEFAULT\_SORT\_COLUMN **=** 'name'

DEFAULT\_SORT\_DIR **=** 'ASC'

# Get all department records.

**def** **self.**get\_all**(**pagenum **=** 1**,** pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

total **=** Department**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

order **=** sort**.**to\_s

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** Department**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get filtered department records.

**def** **self.**get\_filter\_by**(**keyword**,** pagenum **=** 1**,**

pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

criteria **=** Department**.**where**(**'name like ?'**,** "%**#{**keyword**}**%"**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

order **=** sort**.**to\_s

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** criteria**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get the validation errors.

**def** **self.**get\_errors**(**errors**)**

**{** :error **=>** 1**,** :errors **=>** errors **}**

**end**

# Get the item message text.

**def** **self.**item\_message**(**keyword**,** pagenum**,** pagesize**)**

total **=** 0

**if** keyword**.**blank?

total **=** Department**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**else**

criteria **=** Department**.**where**(**'name like ?'**,** "%**#{**keyword**}**%"**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**end**

**end**

**end**

**E:\workspace\payroll\_rails\app\helpers\designation\_helper.rb**

**module** **DesignationHelper**

DEFAULT\_SORT\_COLUMN **=** 'title'

DEFAULT\_SORT\_DIR **=** 'ASC'

# Get all designation records.

**def** **self.**get\_all**(**pagenum **=** 1**,** pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

total **=** Designation**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

order **=** sort**.**to\_s

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** Designation**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get filtered designation records.

**def** **self.**get\_filter\_by**(**find**,** keyword**,** pagenum **=** 1**,**

pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

criteria**,** order **=** get\_filter\_criteria**(**find**,** keyword**,** sort**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

order **=** sort**.**to\_s

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** criteria**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get the validation errors.

**def** **self.**get\_errors**(**errors**)**

**{** :error **=>** 1**,** :errors **=>** errors **}**

**end**

# Get the item message text.

**def** **self.**item\_message**(**keyword**,** pagenum**,** pagesize**)**

total **=** 0

**if** keyword**.**blank?

total **=** Designation**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**else**

criteria **=** Designation**.**where**(**'title like ?'**,** "%**#{**keyword**}**%"**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**end**

**end**

private

**def** **self.**get\_filter\_criteria**(**find**,** keyword**,** sort **=** **nil)**

text **=** "%**#{**keyword**}**%"

order **=** sort**.**present? **?** sort**.**to\_s : **nil**

criteria **=** Designation

**case** find

**when** 1

criteria **=** criteria**.**where**(**'title like ?'**,** text**)**

**return** criteria**,** order

**when** 2

criteria **=** criteria**.**where**(**'`desc` like ?'**,** text**)**

**return** criteria**,** order

**when** 3

criteria **=** criteria**.**where**(**'note like ?'**,** text**)**

**return** criteria**,** order

**else**

criteria **=** criteria**.**where**(**'title like ? or `desc` like ? or note like ?'**,** text**,**

text**,** text**)**

**return** criteria**,** order

**end**

**end**

**end**

**E:\workspace\payroll\_rails\app\helpers\employee\_contact\_helper.rb**

**module** **EmployeeContactHelper**

# Get the validation errors.

**def** **self.**get\_errors**(**errors**)**

**{** :error **=>** 1**,** :errors **=>** errors **}**

**end**

# Checks whether any contact parameters are present in the POST request.

**def** **self.**is\_empty\_params?**(**params**)**

q **=** params**[**:employee\_contact**]**

**if** q**[**:address\_1**].**blank? **&&** q**[**:address\_2**].**blank? **&&** q**[**:address\_3**].**blank? **&&**

q**[**:city**].**blank? **&&** q**[**:state**].**blank? **&&** q**[**:postcode**].**blank? **&&** q**[**:country**].**blank? **&&**

q**[**:home\_phone**].**blank? **&&** q**[**:mobile\_phone**].**blank? **&&** q**[**:work\_email**].**blank? **&&**

q**[**:other\_email**].**blank?

**return** **true**

**end**

**false**

**end**

# Get the contact object from the POST request.

**def** **self.**employee\_contact\_obj**(**o**,** params**)**

q **=** params**[**:employee\_contact**]**

EmployeeContact**.**new**(**:id **=>** o**.**id**,** :address\_1 **=>** q**[**:address\_1**],**

:address\_2 **=>** q**[**:address\_2**],** :address\_3 **=>** q**[**:address\_3**],**

:city **=>** q**[**:city**],** :state **=>** q**[**:state**],** :postcode **=>** q**[**:postcode**],**

:country **=>** q**[**:country**],** :home\_phone **=>** q**[**:home\_phone**],**

:mobile\_phone **=>** q**[**:mobile\_phone**],** :work\_email **=>** q**[**:work\_email**],**

:other\_email **=>** q**[**:other\_email**])**

**end**

# Update the contact object.

**def** **self.**update\_obj**(**o**,** params**)**

q **=** params**[**:employee\_contact**]**

o**.**update\_attributes**(**:address\_1 **=>** q**[**:address\_1**],** :address\_2 **=>** q**[**:address\_2**],**

:address\_3 **=>** q**[**:address\_3**],** :city **=>** q**[**:city**],**

:state **=>** q**[**:state**],** :postcode **=>** q**[**:postcode**],**

:country **=>** q**[**:country**],** :home\_phone **=>** q**[**:home\_phone**],**

:mobile\_phone **=>** q**[**:mobile\_phone**],** :work\_email **=>** q**[**:work\_email**],**

:other\_email **=>** q**[**:other\_email**])**

**end**

# Get the contact object.

**def** **self.**find**(**id**)**

o **=** **nil**

**begin**

o **=** EmployeeContact**.**find**(**id**)**

**rescue** Exception **=>** e

o **=** EmployeeContact**.**new

**end**

o

**end**

**end**

**E:\workspace\payroll\_rails\app\helpers\employee\_helper.rb**

require 'securerandom'

**module** **EmployeeHelper**

DEFAULT\_SORT\_COLUMN **=** 'staff\_id'

DEFAULT\_SORT\_DIR **=** 'ASC'

# Get all employee records.

**def** **self.**get\_all**(**pagenum **=** 1**,** pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

total **=** Employee**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

order **=** sort**.**to\_s

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

**if** sort**.**column **==** 'd.title' **||** sort**.**column **==** 'es.name' **||**

sort**.**column **==** 'dept.name' **||** sort**.**column **==** 'e.first\_name'

criteria **=** get\_join**({},** sort**)**

**else**

criteria **=** Employee

**end**

list **=** criteria**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get filtered employee records.

**def** **self.**get\_filter\_by**(**filters**,** pagenum **=** 1**,**

pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

criteria**,** order **=** get\_filter\_criteria**(**filters**,** sort**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** criteria**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get the validation errors.

**def** **self.**get\_errors**(**errors**)**

**{** :error **=>** 1**,** :errors **=>** errors **}**

**end**

# Get the item message text.

**def** **self.**item\_message**(**filters**,** pagenum**,** pagesize**)**

total **=** 0

**if** filters**.**blank?

total **=** Employee**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**else**

criteria**,** order **=** get\_filter\_criteria**(**filters**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**end**

**end**

# Get the employee object from the POST request.

**def** **self.**employee\_obj**(**params**)**

q **=** params**[**:employee**]**

\_dob **=** q**[**:dob**]**

dob **=** Date**.**strptime**(**\_dob**,** ApplicationHelper**.**date\_fmt**)** if \_dob**.**present?

Employee**.**new**(**:id **=>** SecureRandom**.**uuid**,** :staff\_id **=>** q**[**:staff\_id**],**

:first\_name **=>** q**[**:first\_name**],** :middle\_name **=>** q**[**:middle\_name**],**

:last\_name **=>** q**[**:last\_name**],** :new\_ic **=>** q**[**:new\_ic**],**

:old\_ic **=>** q**[**:old\_ic**],** :passport\_no **=>** q**[**:passport\_no**],**

:gender **=>** q**[**:gender**],** :marital\_status **=>** q**[**:marital\_status**],**

:nationality **=>** q**[**:nationality**],** :dob **=>** dob**,**

:place\_of\_birth **=>** q**[**:place\_of\_birth**],** :race **=>** q**[**:race**],**

:religion **=>** q**[**:religion**],** :is\_bumi **=>** q**[**:is\_bumi**],**

:user\_id **=>** q**[**:user\_id**])**

**end**

**def** **self.**update\_obj**(**o**,** params**)**

q **=** params**[**:employee**]**

\_dob **=** q**[**:dob**]**

dob **=** Date**.**strptime**(**\_dob**,** ApplicationHelper**.**date\_fmt**)** if \_dob**.**present?

o**.**update\_attributes**(**:staff\_id **=>** q**[**:staff\_id**],** :first\_name **=>** q**[**:first\_name**],**

:middle\_name **=>** q**[**:middle\_name**],** :last\_name **=>** q**[**:last\_name**],**

:new\_ic **=>** q**[**:new\_ic**],** :old\_ic **=>** q**[**:old\_ic**],**

:passport\_no **=>** q**[**:passport\_no**],** :gender **=>** q**[**:gender**],**

:marital\_status **=>** q**[**:marital\_status**],**

:nationality **=>** q**[**:nationality**],** :dob **=>** dob**,**

:place\_of\_birth **=>** q**[**:place\_of\_birth**],** :race **=>** q**[**:race**],**

:religion **=>** q**[**:religion**],** :is\_bumi **=>** q**[**:is\_bumi**],**

:user\_id **=>** q**[**:user\_id**])**

**end**

# Update the employee object.

**def** **self.**update\_info**(**o**,** params**)**

q **=** params**[**:employee**]**

\_dob **=** q**[**:dob**]**

dob **=** Date**.**strptime**(**\_dob**,** ApplicationHelper**.**date\_fmt**)** if \_dob**.**present?

o**.**update\_attributes**(**:first\_name **=>** q**[**:first\_name**],** :middle\_name **=>** q**[**:middle\_name**],**

:last\_name **=>** q**[**:last\_name**],** :new\_ic **=>** q**[**:new\_ic**],**

:old\_ic **=>** q**[**:old\_ic**],** :passport\_no **=>** q**[**:passport\_no**],**

:gender **=>** q**[**:gender**],** :marital\_status **=>** q**[**:marital\_status**],**

:nationality **=>** q**[**:nationality**],** :dob **=>** dob**,**

:place\_of\_birth **=>** q**[**:place\_of\_birth**],** :race **=>** q**[**:race**],**

:religion **=>** q**[**:religion**],** :is\_bumi **=>** q**[**:is\_bumi**])**

**end**

private

**def** **self.**get\_filter\_criteria**(**filters**,** sort **=** **nil)**

employee\_keyword **=** "%**#{**filters**[**:employee**]}**%"

staff\_id\_keyword **=** "%**#{**filters**[**:staff\_id**]}**%"

supervisor\_keyword **=** "%**#{**filters**[**:supervisor**]}**%"

order **=** sort**.**present? **?** sort**.**to\_s : **nil**

**if** filters**[**:employment\_status**]** **!=** 0 **||** filters**[**:designation**]** **!=** 0 **||**

filters**[**:dept**]** **!=** 0 **||** sort**.**present?

criteria **=** get\_join**(**filters**,** sort**)**

**end**

criteria **=** Employee if criteria**.**blank?

**if** filters**[**:employee**].**present?

criteria **=** criteria**.**where**(**

'first\_name like ? or middle\_name like ? or last\_name like ?'**,**

employee\_keyword**,** employee\_keyword**,** employee\_keyword**)**

**end**

**if** filters**[**:staff\_id**].**present?

criteria **=** criteria**.**where**(**'staff\_id like ?'**,** staff\_id\_keyword**)**

**end**

**if** filters**[**:employment\_status**]** **!=** 0

criteria **=** criteria**.**where**(**'es.id = ?'**,** filters**[**:employment\_status**])**

**end**

**if** filters**[**:designation**]** **!=** 0

criteria **=** criteria**.**where**(**'d.id = ?'**,** filters**[**:designation**])**

**end**

**if** filters**[**:dept**]** **!=** 0

criteria **=** criteria**.**where**(**'dept.id = ?'**,** filters**[**:dept**])**

**end**

**return** criteria**,** order

**end**

**def** **self.**get\_join**(**filters**,** sort **=** **nil)**

joinhash **=** **{}**

**if** filters**.**any?

**if** filters**[**:employment\_status**]** **!=** 0

q **=** Employee**.**joins**(**'inner join employee\_job ej on employee.id = ej.id'**)**

**.**joins**(**

'inner join employment\_status es on ej.employment\_status\_id = es.id'**)**

joinhash**[**:employment\_status**]** **=** **true**

**end**

**if** filters**[**:designation**]** **!=** 0

**if** q**.**blank?

q **=** Employee**.**joins**(**'inner join employee\_job ej on employee.id = ej.id'**)**

**.**joins**(**'inner join designation d on ej.designation\_id = d.id'**)**

**else**

q **=** q**.**joins**(**'inner join designation d on ej.designation\_id = d.id'**)**

**end**

joinhash**[**:designation**]** **=** **true**

**end**

**if** filters**[**:dept**]** **!=** 0

**if** q**.**blank?

q **=** Employee**.**joins**(**'inner join employee\_job ej on employee.id = ej.id'**)**

**.**joins**(**'inner join department dept on ej.department\_id = dept.id'**)**

**else**

q **=** q**.**joins**(**'inner join department dept on ej.department\_id = dept.id'**)**

**end**

joinhash**[**:dept**]** **=** **true**

**end**

**end**

**if** sort**.**present?

**if** sort**.**column **==** 'd.title'

**if** joinhash**.**has\_key?**(**:employment\_status**)** **&&** **!**joinhash**.**has\_key?**(**:designation**)**

q **=** q**.**joins**(**'left outer join designation d on ej.designation\_id = d.id'**)**

**elsif** **!**joinhash**.**has\_key?**(**:employment\_status**)** **&&** **!**joinhash**.**has\_key?**(**:designation**)**

**if** q**.**blank?

q **=** Employee**.**joins**(**'left outer join employee\_job ej on employee.id = ej.id'**)**

**.**joins**(**'left outer join designation d on ej.designation\_id = d.id'**)**

**else**

q **=** q**.**joins**(**'left outer join employee\_job ej on employee.id = ej.id'**)**

**.**joins**(**'left outer join designation d on ej.designation\_id = d.id'**)**

**end**

**end**

**end**

**if** sort**.**column **==** 'es.name'

**if** **!**joinhash**.**has\_key?**(**:employment\_status**)**

**if** q**.**blank?

q **=** Employee**.**joins**(**'left outer join employee\_job ej on employee.id = ej.id'**)**

**.**joins**(**

'left outer join employment\_status es on ej.employment\_status\_id = es.id'**)**

**else**

q **=** q**.**joins**(**'left outer join employee\_job ej on employee.id = ej.id'**)**

**.**joins**(**

'left outer join employment\_status es on ej.employment\_status\_id = es.id'**)**

**end**

**end**

**end**

**end**

**if** sort**.**column **==** 'dept.name'

**if** joinhash**.**has\_key?**(**:employment\_status**)** **&&** **!**joinhash**.**has\_key?**(**:dept**)**

q **=** q**.**joins**(**'left outer join department dept on ej.department\_id = dept.id'**)**

**elsif** **!**joinhash**.**has\_key?**(**:employment\_status**)** **&&** **!**joinhash**.**has\_key?**(**:dept**)**

**if** q**.**blank?

q **=** Employee**.**joins**(**'left outer join employee\_job ej on employee.id = ej.id'**)**

**.**joins**(**

'left outer join department dept on ej.department\_id = dept.id'**)**

**else**

q **=** q**.**joins**(**'left outer join employee\_job ej on employee.id = ej.id'**)**

**.**joins**(**'left outer join department dept on ej.department\_id = dept.id'**)**

**end**

**end**

**end**

q

**end**

**end**

**E:\workspace\payroll\_rails\app\helpers\employee\_job\_helper.rb**

**module** **EmployeeJobHelper**

# Get the validation errors.

**def** **self.**get\_errors**(**errors**)**

**{** :error **=>** 1**,** :errors **=>** errors **}**

**end**

# Checks whether any job parameters are present in the POST request.

**def** **self.**is\_empty\_params?**(**params**)**

q **=** params**[**:employee\_job**]**

**if** q**[**:designation\_id**]** **==** '0' **&&** q**[**:department\_id**]** **==** '0' **&&**

q**[**:employment\_status\_id**]** **==** '0' **&&** q**[**:job\_category\_id**]** **==** '0' **&&**

q**[**:join\_date**].**blank? **&&** q**[**:confirm\_date**].**blank?

**return** **true**

**end**

**false**

**end**

# Get the job object from the POST request.

**def** **self.**employee\_job\_obj**(**o**,** params**)**

q **=** params**[**:employee\_job**]**

\_join\_date **=** q**[**:join\_date**]**

\_confirm\_date **=** q**[**:confirm\_date**]**

join\_date **=** Date**.**strptime**(**\_join\_date**,**

ApplicationHelper**.**date\_fmt**)** if \_join\_date**.**present?

confirm\_date **=** Date**.**strptime**(**\_confirm\_date**,**

ApplicationHelper**.**date\_fmt**)** if \_confirm\_date**.**present?

EmployeeJob**.**new**(**:id **=>** o**.**id**,** :designation\_id **=>** q**[**:designation\_id**],**

:department\_id **=>** q**[**:department\_id**],**

:employment\_status\_id **=>** q**[**:employment\_status\_id**],**

:job\_category\_id **=>** q**[**:job\_category\_id**],** :join\_date **=>** join\_date**,**

:confirm\_date **=>** confirm\_date**)**

**end**

# Update the job object.

**def** **self.**update\_obj**(**o**,** params**)**

q **=** params**[**:employee\_job**]**

\_join\_date **=** q**[**:join\_date**]**

\_confirm\_date **=** q**[**:confirm\_date**]**

join\_date **=** Date**.**strptime**(**\_join\_date**,**

ApplicationHelper**.**date\_fmt**)** if \_join\_date**.**present?

confirm\_date **=** Date**.**strptime**(**\_confirm\_date**,**

ApplicationHelper**.**date\_fmt**)** if \_confirm\_date**.**present?

o**.**update\_attributes**(**:designation\_id **=>** q**[**:designation\_id**],**

:department\_id **=>** q**[**:department\_id**],**

:employment\_status\_id **=>** q**[**:employment\_status\_id**],**

:job\_category\_id **=>** q**[**:job\_category\_id**],** :join\_date **=>** join\_date**,**

:confirm\_date **=>** confirm\_date**)**

**end**

# Get the job object.

**def** **self.**find**(**id**)**

o **=** **nil**

**begin**

o **=** EmployeeJob**.**find**(**id**)**

**rescue** Exception **=>** e

o **=** EmployeeJob**.**new

**end**

o

**end**

**end**

**E:\workspace\payroll\_rails\app\helpers\employee\_qualification\_helper.rb**

**module** **EmployeeQualificationHelper**

# Get the validation errors.

**def** **self.**get\_errors**(**errors**)**

**{** :error **=>** 1**,** :errors **=>** errors **}**

**end**

# Checks whether any qualification parameters are present in the POST request.

**def** **self.**is\_empty\_params?**(**params**)**

q **=** params**[**:employee\_qualification**]**

**if** q**[**:level**]** **==** '0' **&&** q**[**:institute**].**blank? **&&** q**[**:major**].**blank? **&&** q**[**:year**].**blank? **&&**

q**[**:gpa**].**blank? **&&** q**[**:start\_date**].**blank? **&&** q**[**:end\_date**].**blank?

**return** **true**

**end**

**false**

**end**

# Get the qualification object from the POST request.

**def** **self.**employee\_qualification\_obj**(**o**,** params**)**

q **=** params**[**:employee\_qualification**]**

\_start\_date **=** q**[**:start\_date**]**

\_end\_date **=** q**[**:end\_date**]**

start\_date **=** Date**.**strptime**(**\_start\_date**,**

ApplicationHelper**.**date\_fmt**)** if \_start\_date**.**present?

end\_date **=** Date**.**strptime**(**\_end\_date**,** ApplicationHelper**.**date\_fmt**)** if \_end\_date**.**present?

EmployeeQualification**.**new**(**:id **=>** o**.**id**,** :level **=>** q**[**:level**],**

:institute **=>** q**[**:institute**],** :major **=>** q**[**:major**],**

:year **=>** q**[**:year**],** :gpa **=>** q**[**:gpa**],**

:start\_date **=>** start\_date**,** :end\_date **=>** end\_date**)**

**end**

# Update the qualification object.

**def** **self.**update\_obj**(**o**,** params**)**

q **=** params**[**:employee\_qualification**]**

\_start\_date **=** q**[**:start\_date**]**

\_end\_date **=** q**[**:end\_date**]**

start\_date **=** Date**.**strptime**(**\_start\_date**,**

ApplicationHelper**.**date\_fmt**)** if \_start\_date**.**present?

end\_date **=** Date**.**strptime**(**\_end\_date**,** ApplicationHelper**.**date\_fmt**)** if \_end\_date**.**present?

o**.**update\_attributes**(**:level **=>** q**[**:level**],** :institute **=>** q**[**:institute**],**

:major **=>** q**[**:major**],** :year **=>** q**[**:year**],** :gpa **=>** q**[**:gpa**],**

:start\_date **=>** start\_date**,** :end\_date **=>** end\_date**)**

**end**

# Get the contact object.

**def** **self.**find**(**id**)**

o **=** **nil**

**begin**

o **=** EmployeeQualification**.**find**(**id**)**

**rescue** Exception **=>** e

o **=** EmployeeQualification**.**new

**end**

o

**end**

**end**

**E:\workspace\payroll\_rails\app\helpers\employee\_salary\_helper.rb**

**module** **EmployeeSalaryHelper**

# Get the validation errors

**def** **self.**get\_errors**(**errors**)**

**{** :error **=>** 1**,** :errors **=>** errors **}**

**end**

# Checks whether any salary parameters are present in the POST request.

**def** **self.**is\_empty\_params?**(**params**)**

q **=** params**[**:employee\_salary**]**

**if** q**[**:salary**].**blank? **&&** q**[**:allowance**].**blank? **&&** q**[**:epf**].**blank? **&&** q**[**:socso**].**blank? **&&**

q**[**:bank\_name**].**blank? **&&** q**[**:bank\_acc\_no**].**blank? **&&** q**[**:bank\_acc\_type**].**blank? **&&**

q**[**:bank\_address**].**blank? **&&** q**[**:epf\_no**].**blank? **&&** q**[**:socso\_no**].**blank? **&&**

q**[**:income\_tax\_no**].**blank?

**return** **true**

**elsif** q**[**:salary**]** **==** '0' **&&** q**[**:allowance**]** **==** '0' **&&** q**[**:epf**]** **==** '0' **&&**

q**[**:socso**]** **==** '0' **&&** q**[**:bank\_name**].**blank? **&&** q**[**:bank\_acc\_no**].**blank? **&&**

q**[**:bank\_acc\_type**].**blank? **&&** q**[**:bank\_address**].**blank? **&&** q**[**:epf\_no**].**blank? **&&**

q**[**:socso\_no**].**blank? **&&** q**[**:income\_tax\_no**].**blank?

**return** **true**

**end**

**false**

**end**

# Get the salary object from the POST request.

**def** **self.**employee\_salary\_obj**(**o**,** params**)**

q **=** params**[**:employee\_salary**]**

EmployeeSalary**.**new**(**:id **=>** o**.**id**,** :salary **=>** q**[**:salary**],** :allowance **=>** q**[**:allowance**],**

:epf **=>** q**[**:epf**],** :socso **=>** q**[**:socso**],**

:income\_tax **=>** q**[**:income\_tax**],** :bank\_name **=>** q**[**:bank\_name**],**

:bank\_acc\_no **=>** q**[**:bank\_acc\_no**],**

:bank\_acc\_type **=>** q**[**:bank\_acc\_type**],**

:bank\_address **=>** q**[**:bank\_address**],** :epf\_no **=>** q**[**:epf\_no**],**

:socso\_no **=>** q**[**:socso\_no**],** :income\_tax\_no **=>** q**[**:income\_tax\_no**],**

:pay\_type **=>** q**[**:pay\_type**])**

**end**

# Update the salary object.

**def** **self.**update\_obj**(**o**,** params**)**

q **=** params**[**:employee\_salary**]**

o**.**update\_attributes**(**:salary **=>** q**[**:salary**],** :allowance **=>** q**[**:allowance**],**

:epf **=>** q**[**:epf**],** :socso **=>** q**[**:socso**],**

:income\_tax **=>** q**[**:income\_tax**],** :bank\_name **=>** q**[**:bank\_name**],**

:bank\_acc\_no **=>** q**[**:bank\_acc\_no**],**

:bank\_acc\_type **=>** q**[**:bank\_acc\_type**],**

:bank\_address **=>** q**[**:bank\_address**],** :epf\_no **=>** q**[**:epf\_no**],**

:socso\_no **=>** q**[**:socso\_no**],** :income\_tax\_no **=>** q**[**:income\_tax\_no**],**

:pay\_type **=>** q**[**:pay\_type**])**

**end**

# Get the salary object.

**def** **self.**find**(**id**)**

o **=** **nil**

**begin**

o **=** EmployeeSalary**.**find**(**id**)**

**rescue** Exception **=>** e

o **=** EmployeeSalary**.**new

**end**

o

**end**

**end**

**E:\workspace\payroll\_rails\app\helpers\employment\_status\_helper.rb**

**module** **EmploymentStatusHelper**

DEFAULT\_SORT\_COLUMN **=** 'name'

DEFAULT\_SORT\_DIR **=** 'ASC'

# Get all employment status records.

**def** **self.**get\_all**(**pagenum **=** 1**,** pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

total **=** EmploymentStatus**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

order **=** sort**.**to\_s

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** EmploymentStatus**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get filtered employment status records.

**def** **self.**get\_filter\_by**(**keyword**,** pagenum **=** 1**,**

pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

criteria **=** EmploymentStatus**.**where**(**'name like ?'**,** "%**#{**keyword**}**%"**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

order **=** sort**.**to\_s

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** criteria**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get the validation errors.

**def** **self.**get\_errors**(**errors**)**

**{** :error **=>** 1**,** :errors **=>** errors **}**

**end**

# Get the item message text.

**def** **self.**item\_message**(**keyword**,** pagenum**,** pagesize**)**

total **=** 0

**if** keyword**.**blank?

total **=** EmploymentStatus**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**else**

criteria **=** EmploymentStatus**.**where**(**'name like ?'**,** "%**#{**keyword**}**%"**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**end**

**end**

**end**

**E:\workspace\payroll\_rails\app\helpers\job\_category\_helper.rb**

**module** **JobCategoryHelper**

DEFAULT\_SORT\_COLUMN **=** 'name'

DEFAULT\_SORT\_DIR **=** 'ASC'

# Get all job category records.

**def** **self.**get\_all**(**pagenum **=** 1**,** pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

total **=** JobCategory**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

order **=** sort**.**to\_s

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** JobCategory**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get filtered job category records.

**def** **self.**get\_filter\_by**(**keyword**,** pagenum **=** 1**,**

pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

criteria **=** JobCategory**.**where**(**'name like ?'**,** "%**#{**keyword**}**%"**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

order **=** sort**.**to\_s

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** criteria**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get the validation errors.

**def** **self.**get\_errors**(**errors**)**

**{** :error **=>** 1**,** :errors **=>** errors **}**

**end**

# Get the item message text.

**def** **self.**item\_message**(**keyword**,** pagenum**,** pagesize**)**

total **=** 0

**if** keyword**.**blank?

total **=** JobCategory**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**else**

criteria **=** JobCategory**.**where**(**'name like ?'**,** "%**#{**keyword**}**%"**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**end**

**end**

**end**

**E:\workspace\payroll\_rails\app\helpers\overtime\_rate\_helper.rb**

**module** **OvertimeRateHelper**

DEFAULT\_SORT\_COLUMN **=** 'year'

DEFAULT\_SORT\_DIR **=** 'ASC'

# Get all overtime rate records.

**def** **self.**get\_all**(**pagenum **=** 1**,** pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

total **=** OvertimeRate**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

order **=** sort**.**to\_s

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** OvertimeRate**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get filtered overtime rate records.

**def** **self.**get\_filter\_by**(**filters**,** pagenum **=** 1**,**

pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

criteria**,** order **=** get\_filter\_criteria**(**filters**,** sort**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** criteria**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get the validation errors.

**def** **self.**get\_errors**(**errors**)**

**{** :error **=>** 1**,** :errors **=>** errors **}**

**end**

# Get the item message text.

**def** **self.**item\_message**(**filters**,** pagenum**,** pagesize**)**

total **=** 0

**if** filters**.**blank?

total **=** OvertimeRate**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**else**

criteria**,** order **=** get\_filter\_criteria**(**filters**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**end**

**end**

private

**def** **self.**get\_filter\_criteria**(**filters**,** sort **=** **nil)**

order **=** sort**.**present? **?** sort**.**to\_s : **nil**

criteria **=** OvertimeRate

**if** filters**[**:year**]** **!=** 0

criteria **=** criteria**.**where**(**:year **=>** filters**[**:year**])**

**end**

**return** criteria**,** order

**end**

**end**

**E:\workspace\payroll\_rails\app\helpers\pay\_rate\_helper.rb**

**module** **PayRateHelper**

DEFAULT\_SORT\_COLUMN **=** 'staff\_id'

DEFAULT\_SORT\_DIR **=** 'ASC'

# Get all pay rate records.

**def** **self.**get\_all**(**pagenum **=** 1**,** pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

total **=** PayRate**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

order **=** sort**.**to\_s

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** PayRate**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get filtered pay rate records.

**def** **self.**get\_filter\_by**(**filters**,** pagenum **=** 1**,**

pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

criteria**,** order **=** get\_filter\_criteria**(**filters**,** sort**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** criteria**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get the validation errors.

**def** **self.**get\_errors**(**errors**)**

**{** :error **=>** 1**,** :errors **=>** errors **}**

**end**

# Get the item message text.

**def** **self.**item\_message**(**filters**,** pagenum**,** pagesize**)**

total **=** 0

**if** filters**.**blank?

total **=** PayRate**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**else**

criteria**,** order **=** get\_filter\_criteria**(**filters**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**end**

**end**

# Get the pay rate for a staff id in the specified year and month.

**def** **self.**get\_pay\_rate**(**filters**)**

o **=** PayRate**.**where**(**:staff\_id **=>** filters**[**:staff\_id**])**

**.**where**(**:year **=>** filters**[**:year**])**

**.**where**(**:month **=>** filters**[**:month**]).**first

rate **=** 0

rate **=** o**.**hourly\_pay\_rate if o**.**present?

rate

**end**

private

**def** **self.**get\_filter\_criteria**(**filters**,** sort **=** **nil)**

staff\_id\_keyword **=** "%**#{**filters**[**:staff\_id**]}**%"

order **=** sort**.**present? **?** sort**.**to\_s : **nil**

criteria **=** PayRate

**if** filters**[**:staff\_id**].**present?

criteria **=** criteria**.**where**(**'staff\_id like ?'**,** staff\_id\_keyword**)**

**end**

**if** filters**[**:month**]** **!=** 0

criteria **=** criteria**.**where**(**:month **=>** filters**[**:month**])**

**end**

**if** filters**[**:year**]** **!=** 0

criteria **=** criteria**.**where**(**:year **=>** filters**[**:year**])**

**end**

**return** criteria**,** order

**end**

**end**

**E:\workspace\payroll\_rails\app\helpers\payslip\_helper.rb**

**module** **PayslipHelper**

# Get the total deductions for one employee.

**def** **self.**total\_deductions**(**employee\_salary**)**

**return** 0.0 if employee\_salary**.**blank?

employee\_salary**.**epf **+** employee\_salary**.**socso **+** employee\_salary**.**income\_tax

**end**

# Get the total earnings for one monthly paid employee.

**def** **self.**total\_earnings**(**employee\_salary**,** adjustment**,** total\_overtime\_earnings**)**

**return** 0.0 if employee\_salary**.**blank?

amt **=** employee\_salary**.**salary **+** adjustment

amt **+** employee\_salary**.**allowance **+** total\_overtime\_earnings

**end**

# Get the nett salary for one monthly paid employee.

**def** **self.**nett\_salary**(**earnings**,** deductions**)**

earnings **-** deductions

**end**

# Get the total earnings for one hourly paid employee.

**def** **self.**total\_earnings\_hourly**(**employee\_salary**,** filters**)**

total\_hours **=** AttendanceHelper**.**get\_total\_hours**(**filters**)**

rate **=** PayRateHelper**.**get\_pay\_rate**(**filters**)**

earnings **=** **(**total\_hours **\*** rate**)** **+** employee\_salary**.**allowance

**return** earnings**,** total\_hours**,** rate

**end**

# Get the nett salary for one hourly paid employee.

**def** **self.**nett\_salary\_hourly**(**earnings**,** deductions**)**

earnings **-** deductions

**end**

# Get the total overtime for a staff id in the specified year and month.

**def** **self.**total\_overtime**(**filters**)**

year **=** filters**[**:year**]**

month **=** filters**[**:month**]**

id **=** filters**[**:staff\_id**]**

list **=** Attendance**.**where**(**:staff\_id **=>** id**)**

**.**where**(**'month(work\_date) = ?'**,** month**)**

**.**where**(**'year(work\_date) = ?'**,** year**).**all

duration **=** 0

list**.**each **do** **|**o**|**

to **=** o**.**time\_out**.**localtime

v **=** Time**.**new**(**to**.**year**,** to**.**month**,** to**.**day**,** 18**,** 0**,** 0**)**

x **=** **(**to **-** v**)** **/** 3600.0

duration **+=** x

**end**

duration

**end**

# Get total overtime earnings for a specified year.

**def** **self.**total\_overtime\_earnings**(**filters**,** duration**)**

year **=** filters**[**:year**]**

o **=** OvertimeRate**.**where**(**:year **=>** year**).**first

total **=** 0

**if** o**.**present?

total **=** **(**duration **/** o**.**duration**)** **\*** o**.**pay\_rate

**end**

total

**end**

**end**

**E:\workspace\payroll\_rails\app\helpers\salary\_adjustment\_helper.rb**

**module** **SalaryAdjustmentHelper**

DEFAULT\_SORT\_COLUMN **=** 'staff\_id'

DEFAULT\_SORT\_DIR **=** 'ASC'

# Get all salary adjustment records.

**def** **self.**get\_all**(**pagenum **=** 1**,** pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

total **=** SalaryAdjustment**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

order **=** sort**.**to\_s

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** SalaryAdjustment**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get filtered salary adjustment records.

**def** **self.**get\_filter\_by**(**filters**,** pagenum **=** 1**,**

pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

criteria**,** order **=** get\_filter\_criteria**(**filters**,** sort**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** criteria**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get the validation errors.

**def** **self.**get\_errors**(**errors**)**

**{** :error **=>** 1**,** :errors **=>** errors **}**

**end**

# Get the item message text.

**def** **self.**item\_message**(**filters**,** pagenum**,** pagesize**)**

total **=** 0

**if** filters**.**blank?

total **=** SalaryAdjustment**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**else**

criteria**,** order **=** get\_filter\_criteria**(**filters**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**end**

**end**

# Get the salary adjustment for a staff id in the specified year.

**def** **self.**get\_salary\_adjustment**(**filters**)**

o **=** SalaryAdjustment**.**where**(**:staff\_id **=>** filters**[**:staff\_id**])**

**.**where**(**'year <= ?'**,** filters**[**:year**])**

**.**sum**(**'inc'**)**

o**.**to\_f

**end**

private

**def** **self.**get\_filter\_criteria**(**filters**,** sort **=** **nil)**

staff\_id\_keyword **=** "%**#{**filters**[**:staff\_id**]}**%"

order **=** sort**.**present? **?** sort**.**to\_s : **nil**

criteria **=** SalaryAdjustment

**if** filters**[**:staff\_id**].**present?

criteria **=** criteria**.**where**(**'staff\_id like ?'**,** staff\_id\_keyword**)**

**end**

**if** filters**[**:month**]** **!=** 0

criteria **=** criteria**.**where**(**:month **=>** filters**[**:month**])**

**end**

**if** filters**[**:year**]** **!=** 0

criteria **=** criteria**.**where**(**:year **=>** filters**[**:year**])**

**end**

**return** criteria**,** order

**end**

**end**

**E:\workspace\payroll\_rails\app\helpers\user\_helper.rb**

**module** **UserHelper**

DEFAULT\_SORT\_COLUMN **=** 'username'

DEFAULT\_SORT\_DIR **=** 'ASC'

# Get all user records.

**def** **self.**get\_all**(**pagenum **=** 1**,** pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

total **=** User**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

order **=** sort**.**to\_s

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** User**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,** :limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get filtered user records.

**def** **self.**get\_filter\_by**(**filters**,** pagenum **=** 1**,**

pagesize **=** ApplicationHelper**::**Pager**.**default\_page\_size**,**

sort **=** ApplicationHelper**::**Sort**.**new**(**DEFAULT\_SORT\_COLUMN**,** DEFAULT\_SORT\_DIR**))**

criteria**,** order **=** get\_filter\_criteria**(**filters**,** sort**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

order **=** sort**.**to\_s

has\_next **=** pager**.**has\_next? **?** 1 : 0

has\_prev **=** pager**.**has\_prev? **?** 1 : 0

list **=** criteria**.**order**(**order**).**all**(**:offset **=>** pager**.**lower\_bound**,**

:limit **=>** pager**.**pagesize**)**

**{** :item\_msg **=>** pager**.**item\_message**,**

:hasnext **=>** has\_next**,**

:hasprev **=>** has\_prev**,**

:nextpage **=>** pager**.**pagenum **+** 1**,**

:prevpage **=>** pager**.**pagenum **-** 1**,**

:list **=>** list**,**

:sortcolumn **=>** sort**.**column**,**

:sortdir **=>** sort**.**direction**,**

:page **=>** pager**.**pagenum**,**

:totalpage **=>** pager**.**total\_pages **}**

**end**

# Get the validation errors.

**def** **self.**get\_errors**(**errors**)**

**{** :error **=>** 1**,** :errors **=>** errors **}**

**end**

# Get the item message text.

**def** **self.**item\_message**(**filters**,** pagenum**,** pagesize**)**

total **=** 0

**if** filters**.**blank?

total **=** User**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**else**

criteria **=** User**.**where**(**'username like ?'**,** "%**#{**keyword**}**%"**)**

total **=** criteria**.**count

pager **=** ApplicationHelper**::**Pager**.**new**(**total**,** pagenum**,** pagesize**)**

**return** pager**.**item\_message

**end**

**end**

private

**def** **self.**get\_filter\_criteria**(**filters**,** sort **=** **nil)**

employee\_keyword **=** "%**#{**filters**[**:employee**]}**%"

username\_keyword **=** "%**#{**filters**[**:username**]}**%"

order **=** sort**.**present? **?** sort**.**to\_s : **nil**

**if** filters**[**:employee**].**present?

criteria **=** get\_join**(**filters**)**

**else**

criteria **=** User

**end**

**if** filters**[**:username**].**present?

criteria **=** criteria**.**where**(**'username like ?'**,** username\_keyword**)**

**end**

**if** filters**[**:role**]** **!=** 0

criteria **=** criteria**.**where**(**:role **=>** filters**[**:role**])**

**end**

**if** filters**[**:employee**].**present?

criteria **=** criteria**.**where**(**

'e.first\_name like ? or e.middle\_name like ? or e.last\_name like ?'**,**

employee\_keyword**,** employee\_keyword**,** employee\_keyword**)**

**end**

**if** filters**[**:status**]** **!=** 0

criteria **=** criteria**.**where**(**:status **=>** filters**[**:status**]** **==** 1 **?** **true** : **false)**

**end**

**return** criteria**,** order

**end**

**def** **self.**get\_join**(**filters**)**

**if** filters**[**:employee**].**present?

User**.**joins**(**'inner join employee e on user.id = e.user\_id'**)**

**end**

**end**

**end**