|  |
| --- |
| **Softura Solutions** |
| Entregable PHP Laravel  Proceso para hacer un proyecto que haga las funciones de un CRUD |

1. **Instalando de Herramientas y Configuración de Base de Datos**

Se necesitan las siguientes herramientas en el equipo:

• Wampserver

• Composer

• Laravel 5.5

• PHP 7.2

• MySQL 5

• Apache 2.4.41

La instalación de Laravel se hace con composer utilizando la línea de comandos del sistema, utilizando el siguiente comando:

composer create-project --prefer-dist laravel/laravel nuevo-proyecto "5.5.\*“

Después tenemos que crear nuestro virtual host utilizando la dirección de nuestra carpeta donde se creó nuestro proyecto:

Gráfico de dispersión

Descripción generada automáticamente con confianza media

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Colocamos la ruta de la carpeta de nuestro proyecto

Gráfico de dispersión

Descripción generada automáticamente con confianza media

Aquí están nuestros servidores virtuales

Damos clic para que se cree nuestro servidor virtual

Colocamos el nombre del servidor virtual

Texto

Descripción generada automáticamente

así se ve nuestra página de bienvenida.

Procedemos a hacer la base de datos con el nombre: curso\_laravel\_0822.

En nuestro idle de desarrollo vamos a: config/database.php, aquí se hace la conexión a la base de datos, puede ser a través de diferentes gestores pero nosotros usaremos MySQL, vamos a connections y en el apartado de MySQL colocamos los datos de nuestra BD (nombre de bd, puerto,username, password, charset y collation).

Captura de pantalla de computadora

Descripción generada automáticamente

Luego hacemos lo mismo pero en el archivo .envInterfaz de usuario gráfica

Descripción generada automáticamente con confianza media

1. **Crear Login y empezamos CRUD**

Abrimos nuestro cmd y colocándonos en la carpeta de nuestro proyecto o en la terminal de nuestro idle colocamos el siguiente comando:

php artisan make:auth

Con esto ya está creado el auth en nuestra carpeta ahora falta hacer las tablas en nuestra bd, esto lo hacemos con el comando:

php artisan migrate

Imagen de la pantalla de un celular con letras

Descripción generada automáticamente con confianza media

Luego creamos nuestra tabla empleado y hacemos la migración de esta.

php artisan make:migration create\_empleado\_table

php artisan migrate

Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Aquí se pueden ver nuestras migraciones en el phpMyAdmin.

Hacemos nuestro modelo de empleado y controlador con el siguiente comando:

php artisan make:model Empleado

php artisan make:controller EmpleadoController

En nuestro modelo en el array $fillable, colocamos los campos de la siguiente manera:

Texto

Descripción generada automáticamente

1. **Front, rutas y funcionalidad del CRUD**

Creamos una carpeta dentro de Resource/views con el nombre Empleado, aquí iran todas las vistas que tengan interacción con empleado.

Crearemos los archivos create.blade.php , index.blade.php, edit.blade.php y show.blade.php dentro de la carpeta empleado.

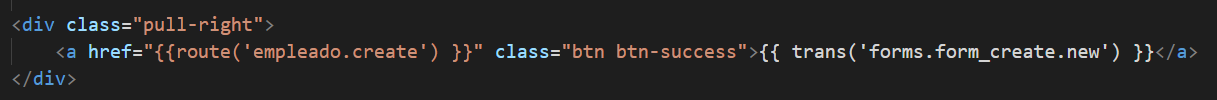
Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

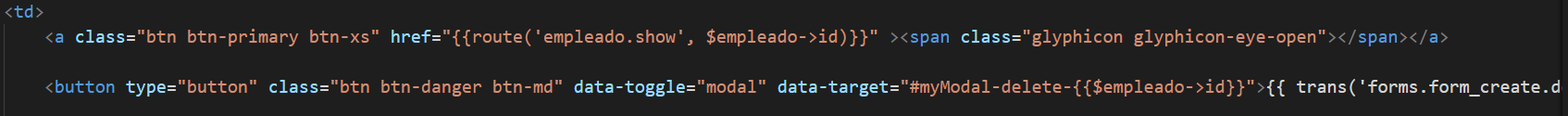
En el index hacemos una tabla en la que se mostraran los empleados que están en registrados en la bd y tendrá un botón para agregar un empleado.

Texto

Descripción generada automáticamente



Dentro de la tabla estarán las acciones de editar, eliminar y de mostrar datos de contacto.





En caso de que no haya datos se muestra el siguiente mensaje.

Texto

Descripción generada automáticamente

La acción de eliminar esta hecha con un modal.

Texto

Descripción generada automáticamente

El propósito de esto es que pregunte si esta seguro de eliminar al empleado seleccionado, la acción de eliminar se hace usando la siguiente parte del código el cual lo envía a las acciones que definimos en el controlador, en este caso la acción de eliminar utilizando la ruta empleado.destroy y pasando el empleado id.



Las rutas se definen en el archivo web que se encuentra en la carpeta routes(routes/web.php).

Captura de pantalla de computadora

Descripción generada automáticamente

Usamos la siguiente ruta para eliminarlo al empleado, la ruta se declara al poner la Route::el método (en este caso delete),entre paréntesis se indica los parámetros que se envían, como la función que se utiliza, después se le coloca un nombre a la ruta para poder ser utilizada mas fácilmente al solo llamarla por su nombre(en este caso llama empleado.destroy).

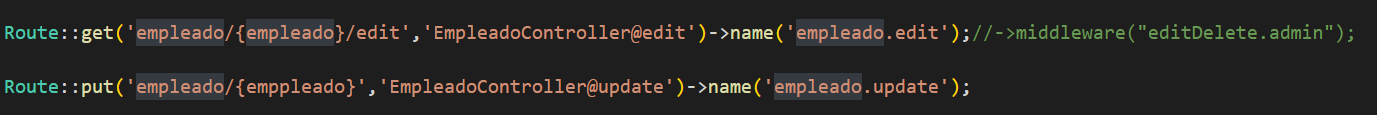


Y en el controlador usamos la función destroy como esta indicada en la ruta, aquí se pide de parámetros el id del empleado, después se indica que se borrara el empleado con base al id recibido y al final nos regresa al index con un mensaje que dice que se eliminó correctamente.

Texto

Descripción generada automáticamente

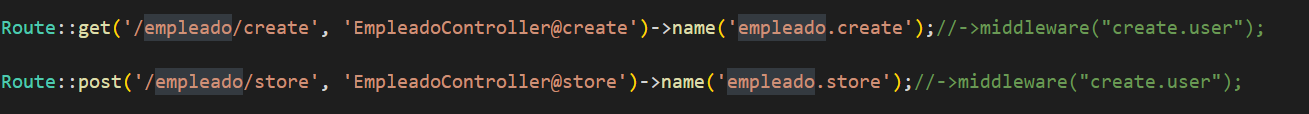
Para editar es una acción similar a eliminar los cambios son que pasa a otra vista donde se editan los datos y luego los validamos para poder ser guardadas.



Texto

Descripción generada automáticamente

Para agregar también se manda a otra ventana en donde se llenan los datos del empleado y después se mandan a su validación, si pasa la validación los guarda y se despliega el nuevo empleado en el index.



Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

1. **Middleware y lenguaje**

Para el middleware se pensó que para hacer las acciones de agregar, editar y eliminar se tiene que iniciar sesión antes, a parte se verifica que sea el administrador el que hará las acciones.

Para esto creamos el archivo EditDeleteByEmail.php el cual verificara que el usuario sea el que indiquemos a través del correo.

Texto

Descripción generada automáticamente

Como funciona es que revisa si el correo del usuario es admin@admin.com (el cual definimos como administrador) y si no eres el usuario que tiene ese correo te direccionara al index cada vez que oprimas alguna de las acciones mencionadas.

Texto

Descripción generada automáticamente

En el kernel agregamos nuestra nueva clase llamada editDelete.admin

Texto

Descripción generada automáticamente

Colocamos el middleware en las rutas de las acciones que mencionamos.

Ahora para la creación de lenguaje, hacemos un archivo en middleware llamado Lenguaje.php.

Texto

Descripción generada automáticamente

El lenguaje nos sirve para colocar la pagina en algún idioma que hallamos definido en este caso es español o ingles.

Captura de pantalla de computadora

Descripción generada automáticamente

En el middleware colocamos la clase hecha de Lenguaje.

Para que se muestre en cualquier ventana que estemos una ves iniciada sesión se hace un dropdown en el archivo app.blade.php que se encuentra en Resource/views/layouts

Captura de pantalla de un celular

Descripción generada automáticamente

Las modificaciones del idioma se hacen el archivo .forms ubicado en diferentes capetas ya que tenemos la carpeta en para ingles y la carpeta es para el español.

Las ubicaciones son resources/lang/es y resources/lang/en

Texto

Descripción generada automáticamente

Y los mismos datos en el archivo en con su respectivos idiomas

Texto

Descripción generada automáticamente

1. **Web service**

Para el web service se uso el tipo de moneda y salario los cuales se implementaron en agregar y editar empleado colocando dos nuevos campos y agregando a la tabla empleado estos dos campos.

Texto

Descripción generada automáticamente

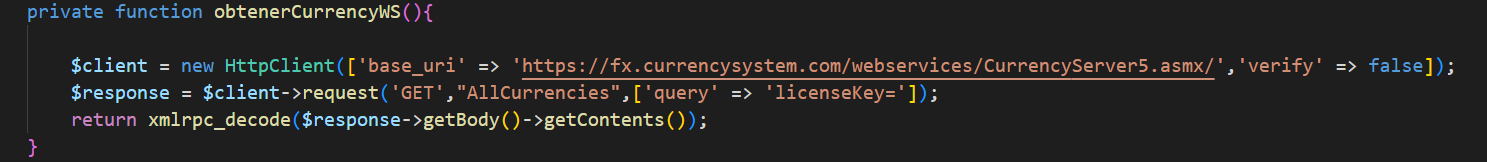
Para esto hacemos la migración de la tabla, la cual al haber sido modificada es un alter\_table con el comando:

php artisan make:migration alter\_empleado\_table

Nesesitamos un servicio llamado Guzzle y para tenerlo ejecutamos el comando:

composer require guzzlehttp/guzzle

Despues para obtener los datos del web service lo hacemos mediante una función hecha en el empleado\_controller llamado obtenerCurrencyWS().



Para que funcione usamos Guzzle y para que este funcione lo importamos en el controlador.



Para su uso en agregar y editar se hace con un arreglo de la siguiente manera.

Texto

Descripción generada automáticamente

Y para su función se colocó de la siguiente manera tanto en agregar como editar.

Texto

Descripción generada automáticamente