Manual del Programador

Al principio tenemos que usar laragon, abrir la terminal y ejecutar el siguiente comando: composer create-project laravel/laravel="7. \*" laravel7 el cual utilice para generar mi proyecto Mueblería, lo siguiente es generar las migraciones con el comando: php artisan make: migration create\_users\_table. Las migraciones de mi proyecto son maderas en donde guardo el tipo de madera del que está hecho el mueble (ej. parota, nogal) la siguiente es la de muebles, en la que guardo el nombre del mueble, la foto, la descripción, el precio, el stock y una llave foránea que es la de maderas que llame madera\_id haciendo referencia al id de maderas, por ultimo tengo la de usuarios que genera por defecto laravel con un comando que explicare más adelante, de esa agregue una variable de tipo integer para separar mis usuarios por tipos, ya se usuario o administrador.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Texto

Descripción generada automáticamente

Utilice seeders para mi proyecto debido a la facilidad de que al iniciar la página ya tenía muebles subidos y un usuario administrador para probar la funcionalidad de la página. El primer seeder es el de tipos de maderas, ahí coloque algunas maderas que son comúnmente utilizadas para la fabricación de los muebles.

Imagen que contiene Gráfico

Descripción generada automáticamente

El otro seeder es el de muebles. Consulte el catálogo de la empresa, de ahí baje las fotos y los detalles del mueble para poder usarlo en los seeders. Estos son algunos de los muebles que tengo en el seeder, utilice de base ocho muebles.

Texto

Descripción generada automáticamente con confianza media

El ultimo seeder es un usuario administrador que deje dentro del sistema para que se puedan probar las funcionalidades de este, ya que no se puede crear una cuenta de administrador, es un administrador sencillo.

Texto

Descripción generada automáticamente

Para que los seeders funcionen es necesario poner la siguiente línea de código dentro del DatabaseSeeder: $this->call (nombredelseeder:class).

Texto

Descripción generada automáticamente

Laravel nos facilita la implementación de usuarios en nuestro proyecto debido a que se pueden generar plantillas de vistas y un controlador, esto se realiza en la terminal de laravel ejecutando los siguientes comandos:

composer require laravel/ui: ^2.4

php artisan ui vue --auth

esto nos genera las vistas necesarias, así como los controladores que laravel tiene por defecto.

 Texto

Descripción generada automáticamente

Dentro del controlador de auth, el RegisterController tuve que realizar unas modificaciones para que funcionara con los usuarios que yo necesitaba, añadí en la función de validator el atributo tipo que sea required e integer, en la función create se añade el tipo.

Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente con confianza baja

En mi pantalla principal del proyecto (welcome.blade.php), separe todo por secciones para que no estuviera todo el código dentro de una sola vista, las partes en que lo separe son: barra de navegación, la galería de muebles y el footer.

Texto

Descripción generada automáticamente

En la primera sección de la barra de navegación es código simple de Bootstrap con la diferencia que añadí a la lista de la barra dos elementos en los que reviso si el usuario es administrador, para que se le desplieguen las opciones de subir un mueble y ver el inventario.

Texto

Descripción generada automáticamente con confianza media

Lo siguiente es que si el usuario esta autenticado aparece la opción de home, donde puede cerrar sesión, en caso de que el usuario no este autenticado (guest) se mostrarán el registro de usuario y el inicio de sesión donde este puede entrar a la página web.

Interfaz de usuario gráfica

Descripción generada automáticamente con confianza media

El usuario sin registrar y el administrador no pueden acceder al carro de compras, esto es exclusivo del usuario, por esto utilice el siguiente código para excluirlos.

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

En la sección de catálogo, envío una variable donde están todos los muebles que hay en la base de datos, para mostrarlos en esta sección. En el controlador, tengo una función llamada index donde hago la asignación de los muebles a la variable. Por último, la ruta es /ver-muebles para que cuando ingrese un usuario pueda ver el catálogo, pero solo ciertos usuarios pueden interactuar con esta galería.

Imagen que contiene Texto

Descripción generada automáticamente



Dentro de la vista hago un foreach para hacer la impresión de cada mueble con sus respectivos campos, para poder mostrar la imagen del mueble en la sección, es necesario colocar específicamente {{asset('storage').'/’. $m->foto}} ya que de otra manera no se mostrará la imagen. También, coloque un botón para que puedas ver los detalles del mueble dentro de un modal, este modal me funciona gracias a un código de JavaScript en donde integro Ajax para poder colocar de manera dinámica los datos del mueble.

Imagen que contiene Texto

Descripción generada automáticamente

Dentro del modal de los detalles, coloque el nombre del mueble en el modal-header una imagen pequeña, la descripción del mueble, el precio, el stock y la madera de la que este hecho. Como especifique anteriormente solo los usuarios pueden agregar muebles al carrito, si el usuario es guest coloque un botón igual que lo redirigirá al login, lo siguiente es que revisa el tipo de usuario que está en la sesión si es tipo 1 significa que es usuario registrado entonces se le mostrará el botón de agregar el carrito.

Imagen que contiene Texto

Descripción generada automáticamente

Texto

Descripción generada automáticamente

El footer es un código HTML que encontré en Bootstrap solo agregué estilos e iconos de Font awesome, así como los enlaces a las redes sociales de la empresa.

Texto

Descripción generada automáticamente

En la vista home es donde el usuario es redirigido cuando inicia sesión o se registra, cuando se inicia sesión aparece un tablero dando la bienvenida al usuario dependiendo del tipo de usuario que sea y dando la bienvenida.

Texto

Descripción generada automáticamente

Las funciones que utilizo para la vista del carrito están dentro de mueblescontroller ya que me parece más eficiente tener todas las funciones en un controlador y ver los errores ahí, como tal el carrito utiliza cuatro funciones.

La función carrito redirige al usuario a la vista carrito donde puede ver sus productos en caso de que los tenga.

Texto

Descripción generada automáticamente con confianza media

La función addcart recibe la variable $mueble de tipo Muebles, entonces revisa si el usuario tiene un carrito, si el usuario no ha añadido nada al carrito simplemente no sale nada en la pantalla, en caso de que el carrito este vacío y se añada un mueble entonces este mueble aparecerá en la vista en una fila de la tabla del carrito, en caso de que quieras añadir más unidades del mismo mueble desde la galería esto solo aumentara la cantidad de tu pedido.

Texto

Descripción generada automáticamente

La función eliminarcart recibe un $id del mueble que queramos borrar, después comprobamos que el mueble este en nuestro carrito, luego hacemos un unset al cart[$id] y redirigimos al usuario a la vista del carrito.

Una captura de pantalla de un celular

Descripción generada automáticamente

Por último tengo la función de cambiarcantidad está la utilizo cuando el usuario desea agregar o quitar un mueble de la cantidad que hay en el carrito de compra esto lo hago con un formulario en la vista en el que tengo dos botones con distinto valor “up” cuando se desea agregar un mueble y “down” cuando deseas quitar uno, esos los tengo dentro de un form post donde envió el $id del mueble, ambos botones son de tipo submit, dentro de la función revisamos si el valor que se envió es “up” o “down”, después si la $id del mueble está definida, además de que si la cantidad es mayor a uno entonces simplemente hacemos un -1 a la cantidad que hay del mueble, en caso de que no directamente llamamos a la función elimnarcart para quitar el mueble. En cambio, cuando el botón es “up” realizamos un proceso similar, con el cambio de que se suma +1 a la cantidad del pedido del mueble. Se redirige a la vista cart.

Texto

Descripción generada automáticamente

La vista del carrito de compras sirve para mostrar los muebles que el usuario quiera comprar se mostrara una tabla con la imagen y el nombre del mueble, el precio del mueble, la cantidad de muebles que el usuario quiera comprar, así como la opción de comprar más o menos, en caso de querer eliminar el mueble se puede usar el botón de quitar y si el mueble es 1 lo elimina o en su defecto al final de la tabla hay una x de eliminar, el subtotal de la compra y el total final de los muebles, utilizo una variable subtotal donde guardo el precio de cada mueble y en caso de que se compren más unidades estas se suman a esa variable, otra llamada total donde sumo el subtotal de los muebles para mostrarlo debajo de la tabla de muebles. Debajo de toda la tabla de muebles y total tengo un botón que redirige al PDF con el desglose del carrito donde puedes ver los muebles y la cantidad que vas a pagar.

Texto

Descripción generada automáticamente

Dentro de la vista PDF tambien utilizo la misma lógica que dentro del carrito, solo que aquí nada más se muestra el mueble y la foto, el precio y el total de todos los muebles que haya pedido. Y por último muestro una imagen para que se pueda realizar el pago.

Texto

Descripción generada automáticamente

En mueblescontroller tengo la función que redirige al PDF es importante saber que este generador de PDF’s es externo a laravel, lo podemos encontrar en github como barrydompdf, a continuación, la liga (<https://github.com/barryvdh/laravel-dompdf>) después de la instalación podemos usar la siguiente función, donde obtengo el carrito del usuario, después compacto los datos del carrito y envío el PDF a una nueva pestaña con el array attachment => 0.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente con confianza media

Las vistas del administrador son la de editar, inventario y crear mueble.

En la vista de crear mueble tengo un formulario POST donde rellenamos el nombre, precio, stock, tipo de madera, la imagen y la descripción. Cuando se sube el mueble aparece un mensaje de éxito.

En el controlador, tengo la función guardamueble al que le envió la request, creo una nueva variable $mueble de tipo Muebles en la que asigno el valor de la request a cada uno de los tipos de valores de $mueble, no es difícil la asignación solo que para guardar la imagen no guardo el path de la imagen si no que guardo la imagen en la carpeta storage y guardo el nombre de la imagen con la función basename, y por último hago un save.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

En el inventario hago un foreach de muebles donde hago lo mismo que en el catálogo donde muestro en una tabla el nombre, precio, imagen, etc. Dentro de esta tabla podemos realizar dos acciones las cuales son editar y eliminar, para estas dos acciones uso un a donde hago un href con el id del mueble.

Texto

Descripción generada automáticamente

Cuando le damos clic en el botón eliminar, en el controlador hay una función que se llama eliminar y recibe como parámetro el id del mueble, hacemos una búsqueda del id con la función find y guardamos el mueble, después borramos el mueble y redirigimos a la ruta ver-muebles.

Texto

Descripción generada automáticamente

En el botón eliminar hacemos algo similar, enviamos el $id del mueble como parámetro y lo igualamos a NULL para después realizar una búsqueda del id con la función find y regreso la ruta edita-muebles con el mueble que quiere editar el administrador.

Texto

Descripción generada automáticamente con confianza baja

En la vista editar mueble podemos ver un formulario similar al que usamos cuando subimos un mueble a la base de datos, la diferencia es que en este formulario en el value ponemos la variable y el campo que queremos ver y cambiar el valor, el action del formulario nos lleva a la ruta edit con el id del mueble.

Texto

Descripción generada automáticamente

La función edit recibe como parámetro la request y el $id del mueble, ahí busco la id del mueble con la función findorfail después hago lo mismo que con la función guardamueble igualo los parámetros de mueble a request, el único cambio es que si sube una imagen nueva al file imagen elimino la que esta almacenada en el storage para no tener dos archivos ahí, por lo que primero elimino el archivo anterior y luego almaceno el nuevo. Por último, guardo el mueble, y los redirijo a la ruta /ver-muebles.

Texto

Descripción generada automáticamente

Las rutas dentro de mi web son las siguientes:

Texto

Descripción generada automáticamente

Las agrupe si están relacionadas, las primeras son relacionadas con la parte inicial de la página web, las otras tienen que ver con el CRUD de muebles, ya que ahí tengo el editar, eliminar, edit.