Laravel / PHP / MySQL

Challenge V4

El objetivo de este desafío consiste en la realización de un gestor (ABM) de Descuentos de una plataforma de renta de vehículos.

Para realizar esta tarea, junto a este documento se entregan:

- El archivo database_ddl.sql, con las sentencias sql de creación de las tablas necesarias para este ejercicio.
- El archivo database_data.sql, con datos para algunas tablas, como brands y regions.
- 2 capturas de pantalla, (list.png y form.png), para tomarse como referncia y guía.

La funcionalidad a desarrollar es la siguiente

Este ABM permite gestionar los descuentos que otorga el sistema cada vez que se realiza la renta de un vehículo.
El descuento, se define por:

- Tipo de acceso al sistema (si es cliente final, agencia o corporativo).
- Por la marca de la rentadora (Avis, Budget, Payless)
- Por la región donde realiza el alquiler del vehículo.
- Los descuentos se pueden definir en función de un código (AWD) que es un dato interno del sistema, conocido por el que carga el descuento, o directamente por un porcentaje de descuento… o por una combinación de ambos.
- Los descuentos se definen según la cantidad de días de alquiler.
  Por ejemplo:
  Entre 1 y 15 días, aplicando el código C12345
  Entre 16 y 25 días, aplicando un descuento de 5%
  Entre 26 y 59 días, una combinación del código C12345 y 5 %.
- Se pueden definir hasta 3 rangos de días, con sus respectivos descuentos.

- **Login de usuario**.
  - Debe requerir el login de un usuario. Estos usuarios deben estar precargados en la base del sistema, o sea, no es necesario realizar el proceso de alta.
  - Si el login es exitoso, pasar directamente al **Listado de Descuentos**.
  - Si el login es fallido, mostrar error y mantenerse en la pantalla de login
  - Usuario a utilizar :
    - Usuario: admin@example.com
    - Clave: password.

- **Formulario de Alta y Edición de Descuentos (form.png)**
  - Se deben validar los datos y su completitud.

- Nombre de la regla (discounts.name): alfanumérico, obligatorio.
    - No se puede repetir en la tabla.
    - Validar tamaño del dato ingresado.
- Activa / Inactiva (discounts.active): 1 = activa / 0 = inactiva
    - por default inactiva.
- Rentadora (discounts.brand_id) obligatorio
    - Sale de la tabla brands.
    - Debe presentar el listado de Marcas.
    - Debe mostrar sólo las marcas activas (brands.active = 1)
    - El listado debe mostrarse ordenado según la columna display_order.
- Tipo de acceso (discounts.access_type_code) obligatorio
    - Sale de la tabla access_types
    - Debe presentar el listado de tipos de acceso.
    - El listado debe mostrarse ordenado según la columna display_order.
- Prioridad (discounts.priority): numérico, obligatorio, default 0.
    - Se puede repetir.
    - Rango válido 1 a 1000
- Región (discounts.region_id): obligatorio
    - Sale de la tabla: regions
    - Debe presentar el listado de Regiones
    - El listado debe mostrarse ordenado según la columna display_order.
- Períodos de Aplicación
    - Se presentan sólo 3 períodos de aplicación.
    - En el alta, se presentan los campos del primer período habilitados y al completarse se habilita el segundo… y luego de completado este se habilita el siguiente.
    - Campos:
        - Desde y hasta: numericos, obligatorios.
            - “Desde” debe ser menor que “Hasta”
            - No es necesario validar que los rangos de días sean consecutivos o que se solapen.
        - Código de Descuento AWD, alfanumérico, opcional.
        - Porcentaje de Descuento GSA, numérico, opcional.
            - Uno de los 2 (código o porcentaje) debe estar completo.
            - Pueden estar los 2.
            - Pero no pueden estar los 2 vacíos.
- Período de aplicación (rango de fechas): obligatorio
    - Corresponde a los campos discounts. start_date y end_date
    - Puede implementarse como 2 campos separados con datepicker.

- **Listado de Descuentos (list.png)**
    - Debe implementar paginado, filtro y ordenamiento.
    - Tener en cuenta que el listado, debe presentar los registros correspondientes a las rentadoras activas únicamente (brands.active == 1)
    - El listado debe presentarse desde el momento en que se presenta la pantalla, con un ordenamiento predeterminado (nombre del descuento).
    - Datos a mostrar, y su correspondiente origen:
        - Rentadora: brands.name

        - Región: regions.name
        - Nombre: discounts.name
        - Tipo de Acceso: access_types.name
        - Estado: discounts.active (1 = activo / 0 = inactivo)
        - Período: registros de discount_ranages: from_days - to_days
        - AWD/BCD: registros de discount_ranges: code
        - Descuento GSA: registros de discount_ranges: discount
        - Período de Promoción: discounts start_date - end_date
        - Prioridad: discounts.priority
    - Tener en cuenta, de las capturas de pantalla, la forma en que se muestran los valores de las columnas (formato de fechas, números y el alineamiento).
    - FIltros del listado:
        - Rentadora
            - Deb presentar la lista de Rentadoras (brands)
            - Debe mostrar sólo las marcas activas (brands.active = 1)
            - El listado debe mostrarse ordenado según la columna display_order.
        - Región
            - Debe presentar el listado de Regiones
            - El listado debe mostrarse ordenado según la columna display_order.
        - Nombre (alfanumerico)
            - Debe buscar este dato en la columna discounts.name (aplicando like)
        - AWD/BCD
            - Debe buscar este dato en la columna discount_ranges.code (debse se exacto)
        - El filtro debe aplicarse en el momento de presionar “Buscar”
    - Exportación de datos
        - Debe descargar un archivo csv, con las mismas columnas del listado, de todos los registros aplicando el filtrado.
        - El nombre del archivo es fijo: descuentos.csv

- **Borrado de descuentos**
    - En el listado de descuentos, se presenta en cada fila un botón para el borrado de descuentos. El borrado debe ser lógico (soft delete).
    - Puede presentarse un modal estándar para confirmar el borrado. (puede hacerse con un confirm() de javascript)

- Multilenguaje:
    - El desarrollo debe implementar internacionalización (i18n). O sea, los textos fijos de las pantallas deben estar traducidos, en Inglés y Español.
    - El idioma a utilizar puede ser configurable por archivo de configuración (.env) (no es necesario que el idioma se cambie desde la pantalla)
    - No son importantes las traducciones en sí, sino cómo se resuelven.

Entregable:

- Para el login de usuarios, puede utilizarse el método estándar que incluye Laravel.

- Subir el proyecto a su github o gitlab **en forma Privada**!

- Agregar como colaborador de ese proyecto a
- El proyecto incluir como mínimo:
    - Las migraciones de todas las tablas.
    - Seeder para
        - la tabla de usuarios (con el usuario indicado más arriba),
        - Las tablas de referencia de las que se entregaron los scripts (inserts)
        - Un set de pruebas, de entre 15 y 30 registros de descuentos.
    - Un documento (puede ser el readme.md) que describa:
        - Requerimientos para el despliegue: principalmente la versión de php y Laravel utilizadas.
        - Paso a paso para desplegar el proyecto en un equipo local. Cada tarea y cada comando a ejecutar, desde que se clona el proyecto, hasta que se levanta el servicio.

Consideraciones generales:

- Las páginas pueden resolverse con Bootstrap y Livewire, u otro framework.
- No se pueden utilizar frameworks como React, Angular, Vue.
- El desarrollo de la funcionalidad de Exportación es opcional.
  Puede utilizarse algún paquete de PHP, como maatwebsite/excel
- El paginado, el filtro y el ordenamiento, deben resolverse del lado del controlador, no por javascript.
- Puntos que serán evaluados:
    - Resolución general de las funcionalidades.
    - Organización y prolijidad del código.
    - Descripción del proceso de despliegue.