**Componente:** HomeComponent

Funcionalidades principales:

El HomeComponent es un componente angular que representa la página de inicio de la aplicación. Es responsable de representar la plantilla HTML y diseñarla con el archivo CSS especificado. También se utiliza para definir el comportamiento de la página de inicio, como el manejo de las interacciones del usuario y la visualización de datos.

**Componente: HeaderComponent**

Funcionalidades principales:

La clase HeaderComponent es responsable de administrar el componente de encabezado de la aplicación. Permite al usuario alternar el campo navbarOpen, que controla si la barra de navegación está abierta o cerrada.

Métodos:

- toggleNavbar(): este método alterna el valor del campo navbarOpen, que controla si la barra de navegación está abierta o cerrada.

Campos:

- navbarOpen: este campo booleano controla si la barra de navegación está abierta o cerrada. Inicialmente se establece en falso.

**Componente VehiclesListComponent:**

Funcionalidades principales:

La clase VehiclesListComponent es responsable de mostrar una lista de marcas de vehículos usando virtual scroll y permitir que el usuario filtre y seleccione una marca para ver más detalles. Recupera la lista de marcas del Store mediante selectores y envía una acción para cargar las marcas si aún no se han cargado. También calcula la altura del contenedor para mostrar la lista y navega a la página de detalles cuando se selecciona una marca.

Métodos:

- ngOnInit(): se suscribe a la carga y hace observables del Store y envía una acción para cargar las marcas

- filterMakes(): filtra la lista de marcas según el campo filterValue

- calculeContainerHeight(): calcula la altura del contenedor en función de la altura de la pantalla

- loadMore(make: MakeModel): envía una acción para seleccionar la marca y navega a la página de detalles

Campos:

- makes$: un observable de la lista de marcas

- loading$: un observable del estado de carga de las marcas

- filterValue: una cadena utilizada para filtrar la lista de marcas

- screenHeight: la altura de la pantalla

**Componente VehiclesDetailComponent:**

Funcionalidades principales:

La clase VehiclesDetailComponent es responsable de mostrar los detalles de una marca de vehículo seleccionada. Recupera el ID de marca de los parámetros de URL y envía acciones para cargar los tipos y modelos de vehículos correspondientes. Si se recarga la página recupera la marca seleccionada del almacenamiento local y la muestra en la página.

Métodos:

- ngOnInit(): se suscribe a los parámetros de URL y envía acciones para cargar tipos y modelos de vehículos. También recupera la marca seleccionada del almacenamiento local y se suscribe a los cambios en la marca seleccionada.

- ngOnDestroy(): elimina la marca seleccionada del almacenamiento local y cancela todas las suscripciones.

Campos:

- makeId: el ID de la marca seleccionada.

- make: el objeto de marca seleccionado que contiene el ID y el nombre de la marca.

- filterValue: una cadena utilizada para filtrar los modelos de vehículos mostrados.

- suscripciones: una matriz de suscripciones que se utiliza para darse de baja de todas las suscripciones cuando se destruye el componente.

**Componente VehiclesTypeComponent:**

Funcionalidades principales:

La clase VehiclesTypeComponent es responsable de mostrar una lista de tipos de vehículos y su estado de carga. Recupera la lista de modelos del estado usando el selector selectListVehiclesType y los muestra en la plantilla. También recupera el estado de carga usando el selector selectVehiclesTypeLoading y lo usa para mostrar una rueda de carga mientras se recuperan los datos.

Métodos:

- ngOnInit(): Este método se llama cuando se inicializa el componente. Selecciona la lista de tipos de vehículos y su estado de carga desde el estado de la aplicación utilizando la biblioteca @ngrx/store y los asigna a los campos vehicleType$ y loading$ respectivamente.

Campos:

- vehicleType$: este campo es un Observable que contiene la lista de tipos de vehículos seleccionados desde el estado de la aplicación.

- loading$: este campo es un Observable que contiene el estado de carga de los tipos de vehículos seleccionados desde el estado de la aplicación.

**Componente** **ModelsMakeComponent**

Funcionalidades principales:

La clase ModelsMakeComponent es responsable de mostrar una lista de modelos y sus respectivas marcas. Recupera la lista de modelos del estado usando el selector selectListModelsMake y los muestra en la plantilla. También recupera el estado de carga usando el selector selectModelsMakeLoading y lo usa para mostrar una rueda de carga mientras se recuperan los datos.

Métodos:

- ngOnInit(): Este método se llama cuando se inicializa el componente. Recupera la lista de modelos del estado y los asigna a los campos correspondientes.

Campos:

- modelsMake$: un Observable que contiene la lista de modelos recuperados del estado.

- loading$: un Observable que contiene el estado de carga recuperado del estado.