

### Requerimientos funcionales

**R#1 Registrar** empleados. Estos empleados serán registrados mediante una llave virtual en el pantalla de registrarse. Los empleados tienen los siguientes atributos: código, nombre, apellido, identificación, nombre de usuario, contraseña, número de ventas, y valor de comisión.

**R#2 Iniciar Sesión.** Cada vez que se cree un empleado este podrá iniciar sesión con su nombre de usuario y contraseña respectivamente, si, la contraseña no coincide con el nombre de usuario y viceversa, saltara una alerta avisando sobre este error.

**R#3 Gestionar** ciudades. Permite gestionar todos los datos de las ciudades, las ciudades tendrán como atributos: código, referencias y ciudad.

- **Añadir** ciudades al sistema con sus atributos correspondientes. Todos son obligatorios.
- **Actualizar** el nombre de las ciudades. Una vez actualizado esto se debe de cambiar los registros en los clientes.
- **Eliminar** las ciudades, solo se puede eliminar si este objeto no está referenciado en un cliente.

**R#4 Gestionar** clientes. Los clientes tendrán los siguientes atributos: código, nombres, cédula, dirección, teléfono, email y ciudad.

- **Añadir** clientes al sistema con sus atributos correspondientes, todos son obligatorios menos el Email, el Email debe estar acompañado por una “@”.
- **Actualizar** todos los datos del clientes excepto el código.
- **Eliminar** un cliente seleccionado para así borrarlo del sistema.

**#5 Gestionar** vehículos. Se podrá gestionar todos los vehículos, los atributos de los vehículos son los siguientes: Código, placa, año, modelo, color, precio por día, foto, disponibilidad, tipo y marca.

- **Añadir** vehículos. Se podrá añadir los vehículos deseados con todos sus atributos debido a que son obligatorios, no se puede repetir el número de placa entre dos vehículos.
- **Actualizar** los vehículos, todos los atributos se pueden actualizar menos el código.
- **Eliminar** los vehículos, se podrá eliminar el vehículo que se desea siempre y cuando no esté ocupado.

**R#6 Gestionar** tipo de vehículos, se podrá gestionar los tipo de vehículos, estos tendrán un código, nombre y referencia.

- **Añadir** tipo de vehículos. Se podrá añadir los tipos de vehículos deseados con todos sus atributos debido a que son obligatorios.
- **Actualizar** los tipos de vehículos, todos los atributos se pueden actualizar menos el código.
- **Eliminar** los tipos de vehículos, se podrá eliminar el tipo de vehículo que se desea siempre y cuando no esté referenciado en otro objeto.

**R#7 Gestionar** marca de vehículos, se podrá gestionar las marcas de vehículos, estos tendrán un código, nombre y referencia.

- **Añadir** marcas de vehículos. Se podrá añadir las marcas de vehículos deseadas con todos sus atributos debido a que son obligatorios.
- **Actualizar** las marcas de vehículos, todos los atributos se pueden actualizar menos el código.
- **Eliminar** las marcas de vehículos, se podrá eliminar la marca de vehículo que se desea siempre y cuando no esté referenciado en otro objeto.

**R#8 Agregar** un alquiler. Se podrá agregar un alquiler al sistema, este alquiler contará con un código, un ticket, un cliente, un vehículo el cual solo se podrá agregar si están disponibles y posteriormente se cambiará de estado, una fecha inicial, una fecha final, cantidad de días, estado del alquiler, atraso, multa, y precio, el cual se calculará multiplicando el precio del carro por la cantidad de días.

**R#9 Devolver** un alquiler. Se podrá devolver un alquiler de vehículo, se pagará el valor dado por la multiplicación de cantidad de días y precio del vehículo. Si el estado del vehículo es atraso, se cobrará una multa, la cual corresponderá al 125% del valor de los días transcurridos de atraso.

**R#10 Listar** todos los vehículos disponibles en el sistema.

**R#11 Listar** todos los clientes disponibles en el sistema.

**R#12 Listar** todas las ciudades disponibles en el sistema.

**R#13 Listar** todos los tipos de vehículos disponibles en el sistema.

**R#14 Listar** todas las marcas de vehículos disponibles en el sistema.

**R#15 Listar** todos los alquileres que se han hecho en el sistema.

**R#16 Mover** los vehículos. Se permite mover con un botón los vehículos en la pantalla donde se muestran esto, cada vez que avanza este botón se actualiza la información de la pantalla.

**R#17 Seleccionar** todos los objetos. Todo objeto que se encuentre en una lista podrá ser seleccionado, en excepción a los empleados debido a que no será necesario.

**R#18 Sumar** comisión. Según el usuario activo cada vez que este haga un alquiler con un cliente, se le sumará a su contador para que así se calcule la comisión correspondiente.

**R#19 Mostrar** los mejores empleados. Se podrá mostrar los mejores empleados en un apartado donde el primer elemento es el empleado con mayor comisión y el ultimo será el de menor comisión.

**R#20 Buscar** clientes. Se buscarán los clientes de acuerdo a su cédula o su nombre respectivamente.

**R#21 Buscar** vehículos. Se podrá buscar los vehículos de acuerdo a su placa registrada.

**R#22 Buscar** ciudades. Se podrá buscar las ciudades de acuerdo a su nombre registrado.

**R#22 Buscar** tipos de vehículos. Se podrá buscar los tipos de vehículo de acuerdo a su nombre registrado.

**R#22 Buscar** marca de vehículos. Se podrá buscar las marcas de acuerdo a su nombre registrado.

**R#22 Buscar** Alquileres. Se podrá buscar los alquileres de acuerdo a su ticket registrado o bien por la cédula del cliente que realizó el alquiler.

**R#23 Importar** clientes. Se podrá importar clientes, este archivo a importar debe ser de tipo csv y sus datos deben de estar separados mediante comas (“,”).

**R#23 Importar** vehículos. Se podrá importar vehículos, este archivo a importar debe ser de tipo csv y sus datos deben de estar separados mediante comas (“,”).

**R#24 Exportar** clientes. Se exportarán todos los clientes o bien filtrar estos clientes por ciudades para así solo exportar los deseados.

**R#25 Exportar** vehículos. Se exportarán todos los vehículos o bien filtrar estos vehículos por tipo de vehículo para así solo exportar los deseados.

**R#26 Exportar** alquileres. Se exportarán todos los vehículos o bien filtrar estos alquileres en un rango de fecha inicial y final, donde solo importa la fecha inicial del alquiler, es

decir, si el inicio de un alquiler está entre el rango estipulado, se imprimirá, y la fecha final no tendrá importancia.

**R#27 Probar** velocidad, cada vez que se visualice los carros estos tendrán la opción la cual nos permitirá ver la velocidad del vehículo para así probarlos de manera virtual.

**R#28 Mostrar** gráfico estadístico (Torta) referente a los estados de las rentas de la aplicación.

**R#29 Mostrar** gráfico estadístico (Barras) referente a la cantidad de objetos creados en la aplicación.

### **Requerimientos no funcionales**

**R#1 Deshabilitar** los paneles cada vez que se abra una ventana, si un botón abre una ventana nueva, el panel que contiene este botón deberá ser deshabilitado.

**R#2 Habilitar** los paneles cada vez que se cierre una pestaña, si un panel se encuentra deshabilitado y la pestaña actual se cierra, el panel anterior debe volverse a habilitar

**R#3 Gestionar** una ciudad, no se podrá crear, actualizar una ciudad con el mismo nombre de otra ciudad, e igualmente no se podrá eliminar una ciudad si esta cuenta con una referencia.

**R#4 Relacionar** personas, las clase persona será una clase padre de la clase empleado y de la clase cliente

**R#5 Gestionar** un cliente, no se podrá crear, actualizar un cliente con el mismo ID de otro cliente, si a la hora de crear o actualizar el cliente, el correo de esto no contiene el carácter “@” tampoco se podrá agregar. Si un cliente no ingresa una ciudad, el programa no dejará que este cliente sea agregado, e igualmente no se podrá eliminar un cliente si esta cuenta con una referencia.

**R#6 Gestionar** una marca, no se podrá crear, actualizar una marca con el mismo nombre de otra marca, e igualmente no se podrá eliminar una marca si esta cuenta con una referencia.

**R#7 Gestionar** un tipo de vehículo, no se podrá crear, actualizar un tipo de vehículo con el mismo nombre de otra marca, e igualmente no se podrá eliminar un tipo de vehículo si este cuenta con una referencia.

**R#8 Gestionar** vehículo, la clase vehículo es la clase padre de la clase carro.

**R#9 Gestionar** un carro, no se podrá crear, actualizar un carro con el mismo número de placa, si un carro no tiene tipo de vehículo y marca tampoco podrá ser creado con éxito, un carro no podrá ser eliminado si cuenta con una referencia hacía otro objeto, e igualmente si el estado es en alquiler.

**R#10 Gestionar** un alquiler, no se podrá crear un alquiler si este no cuenta con un cliente y un vehículo, si un alquiler no es devuelto a tiempo se empezará a cobrar una multa por cada día que pase la cual corresponderá al 125% del valor diario del vehículo.

**R#11 Mostrar estado** de devolución, a la hora de estar en la pantalla de devolución se mostrar una línea de tiempo donde nos mostrara el proceso de devolución, el primer paso es buscando alquiler, el segundo es alquiler seleccionado y el ultimo es alquiler pagado.

**R#12 Mostrar** velocidad del carro, si un carro no posee una foto, este no podrá mostrar la velocidad en la pantalla para testearla.

**R#13 Entrar** a una pantalla donde se gestione algún objeto implicara que, solo estará habilitado el botón de crear tal objeto, si se selecciona ese botón solo se habilitara un el botón de guardar, y si se selecciona un objeto de una lista, los botones de remover y actualizar se habilitaran.

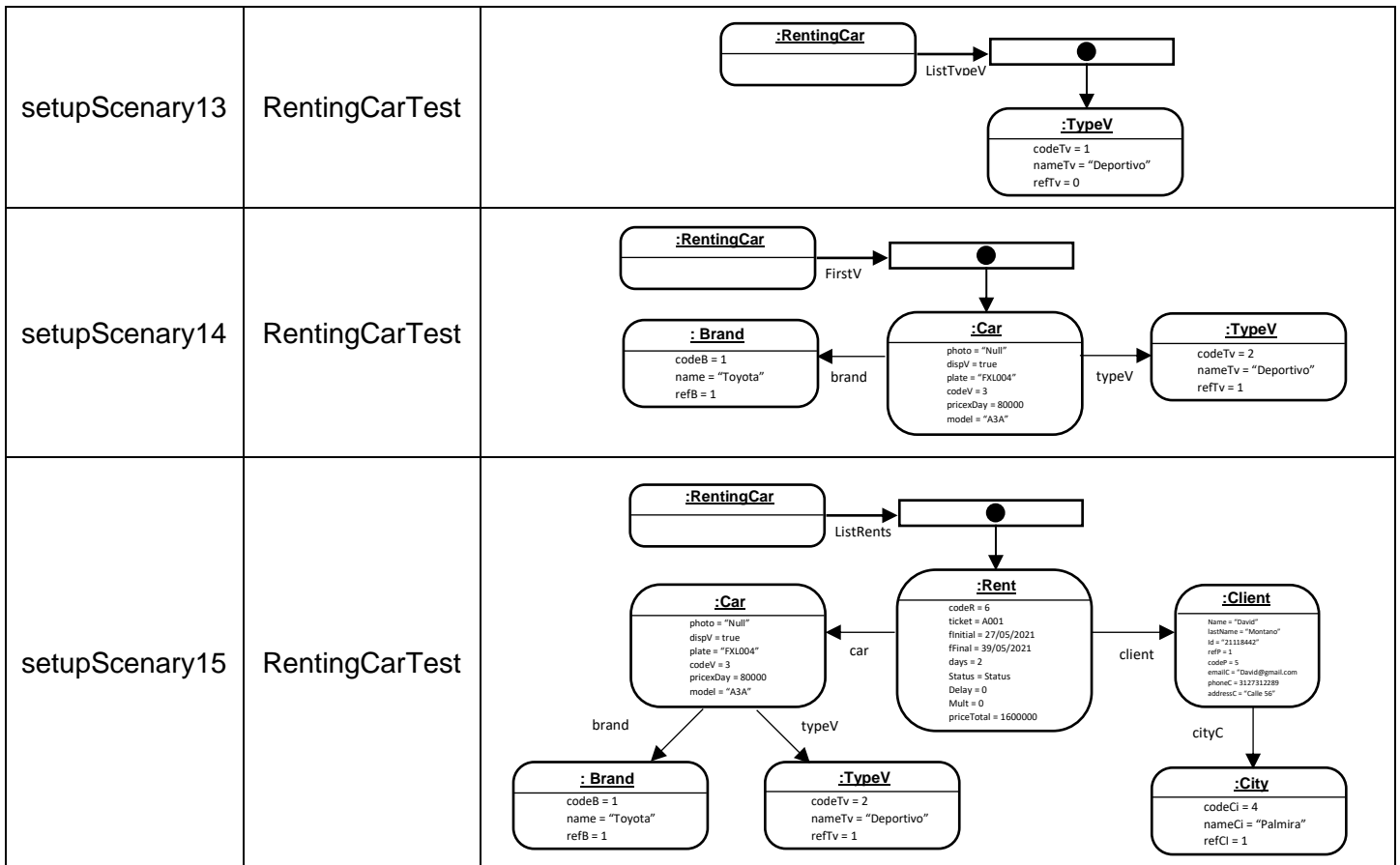
**R#14 Importar** solo se podrán importar archivos CSV con un formato establecido.

## DISEÑO DE PRUEBAS UNITARIAS

## RENTING CAR

## Configuración de escenarios.

Nombre	Clase	Escenario
setupScenary1	ClientTest	Vacío
setupScenary2	EmployeeTest	Vacío
setupScenary3	CityTest	Vacío
setupScenary4	BrandTest	Vacío
setupScenary5	TypeVTest	Vacío
setupScenary6	RentTest	Vacío
setupScenary7	CarTest	Vacío
setupScenary8	RentingCarTest	
setupScenary9	RentingCarTest	
setupScenary10	RentingCarTest	
setupScenary11	RentingCarTest	
setupScenary12	RentingCarTest	



## Diseños de caso de prueba.

Objetivo de la prueba: Validar la correcta creación de un cliente.

Clase	Método	Escenario	Valores de entrada	Resultado
ClientTest	Client	setupScenary1	name = "David" lastName = "Montaña" id = 21118442 refP = 0 codeP = 2 emailC = "David@gmail.com" phone = 3127312289 addressC = "Calle 56" <b>Añadir Ciudad</b> codeCi = 1 nameCi = "Palmira" refCi = 1	Se ha creado un nuevo cliente de forma exitosa. Cada uno de los atributos del nuevo cliente tiene asignada correctamente la información pasada por parámetro.

Objetivo de la prueba: Validar la correcta creación de un empleado

Clase	Método	Escenario	Valores de entrada	Resultado
EmployeeTest	Employee	setupScenary2	name = "Gabriel" lastName = "Suárez" id = 1006325694 refP = 0	Se ha creado un nuevo empleado de forma exitosa. Cada uno de los atributos del nuevo empleado tiene

			codeP = 1 vComision = 0 nSold = 0 password = "gabriels" username = "gabsua"	asignada correctamente la información pasada por parámetro.
--	--	--	---	---

**Objetivo de la prueba:** Validar la correcta creación de una ciudad.

Clase	Método	Escenario	Valores de entrada	Resultado
CityTest	City	setupScenary3	codeCi = 1 nameCi = "Palmira" refCI = 0	Se ha creado una nueva ciudad de forma exitosa. Cada uno de los atributos de la nueva ciudad tiene asignada correctamente la información pasada por parámetro.

**Objetivo de la prueba:** Validar la correcta creación de una marca.

Clase	Método	Escenario	Valores de entrada	Resultado
BrandTest	Brand	setupScenary4	country = "Alemania" codeB = 0 name = "Toyota" refB = 0	Se ha creado una nueva marca de forma exitosa. Cada uno de los atributos de la nueva marca tiene asignada correctamente la información pasada por parámetro.

**Objetivo de la prueba:** Validar la correcta creación de un tipo de vehículo.

Clase	Método	Escenario	Valores de entrada	Resultado
TypeVTest	TypeV	setupScenary5	quality = 100 codeTv = 1 nameTv = "Deportivo" refTv = 0	Se ha creado un nuevo tipo de vehículo de forma exitosa. Cada uno de los atributos del nuevo tipo tiene asignada correctamente la información pasada por parámetro.

**Objetivo de la prueba:** Validar la correcta creación de una renta.

Clase	Método	Escenario	Valores de entrada	Resultado
RentTest	Brand	setupScenary6	codeR = 6 ticket = 1 Finitial = "27/05/2021" Ffinal = "10/06/2021" days = 14 status = DEFERRED delay = 0 mult = 0 priceTotal = 1120000 <b>Añadir marca</b> country = "Alemania" codeB = 1 nameB = "Toyota" refB = 1; <b>Añadir tipo de vehículo</b> quality = 100 codeTv = 2 nameTv = "Deportivo" refTv = 1 <b>Añadir vehículo</b>	Se ha creado una nueva renta de forma exitosa. Cada uno de los atributos de la nueva renta tiene asignada correctamente la información pasada por parámetro.



			model = "A3A" color = "Verde" pricexDay = 800000 codeV = 3 plate = "FXL004" dispV = true photo = "" year = 2021 refV = 1 <b>Añadir cliente</b> name = "David" lastName = "Montaña" id = 21118442 refP = 0 codeP = 2 emailC = "David@gmail.com" phone = 3127312289 addressC = "Calle 56" <b>Añadir ciudad</b> codeCi = 4; nameCi = "Palmira" refCi = 1	
--	--	--	---	--

**Objetivo de la prueba:** Validar la correcta creación de una carro.

Clase	Método	Escenario	Valores de entrada	Resultado
CarTest	Car	setupScenary7	model = "A3A" color = "Verde" pricexDay = 800000 codeV = 3 plate = "FXL004" dispV = true photo = "" year = 2021 refV = 0 <b>Añadir marca</b> country = "Alemania" codeB = 1 nameB = "Toyota" refB = 1; <b>Añadir tipo de vehículo</b> quality = 100 codeTv = 2 nameTv = "Deportivo" refTv = 1	Se ha creado un nuevo carro de forma exitosa. Cada uno de los atributos del nuevo carro tiene asignada correctamente la información pasada por parámetro.

**Objetivo de la prueba:** Validar la correcta creación del RentingCar.

Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	RentingCar	setupScenary8	Ninguno	Se ha creado un nuevo RentingCar de manera exitosa con sus relaciones vacías.

<b>Objetivo de la prueba:</b> Validar la correcta creación de una empleado en el elemento siguiente.				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	addEmp	setupScenary9	Name = "David" lastName = "Montaña" Id = "21118442" refP = "0" codeP = "2" vComision = "0" nSold = "0" password = "davidm" username = "davmon"	Se ha creado el empleado correctamente con sus atributos respectivos pasados como parámetros, en la posición siguiente del empleado firstE, y el método debe de retornar true.

<b>Objetivo de la prueba:</b> Validar que funcione correctamente el Login del empleado				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	loginEmployee	setupScenary9	username = "gabriels" password = "gabsua"	Se ha retornado un valor booleano true, debido a que el Login fue exitoso.

<b>Objetivo de la prueba:</b> Validar la correcta creación de una ciudad.				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	addCity	setupScenary10	codeCi = 2 nameCi = "Tulua" refCI = 0	Se ha creado una nueva ciudad, el método devuelve true, además de que la lista de las ciudades ahora es de tamaño 2, la ciudad cuenta con sus atributos correctamente implementados.

<b>Objetivo de la prueba:</b> Validar que NO se agregue una ciudad con el mismo nombre				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	addCity	setupScenary10	codeCi = 2 nameCi = "Palmira" refCI = 0	No se ha agregado la ciudad debido a que existe otra con el mismo nombre, el método devuelve false, y la lista sigue siendo de tamaño 1.

<b>Objetivo de la prueba:</b> Validar que se actualice una ciudad				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	updateCity	setupScenary10	nameCi = "Tulua"	Se ha actualizado correctamente el nombre de la ciudad, debido a que el nombre no está repetido, devuelve true.

<b>Objetivo de la prueba:</b> Validar que NO se actualice una ciudad debido a que el nombre esta repetido				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	uptadeCity	setupScenary10	nameCi = "Palmira"	No se ha actualizado correctamente el nombre de la ciudad, debido a que ya existe otra con el mismo nombre, devuelve false.

<b>Objetivo de la prueba:</b> Validar que se borre una ciudad de la lista de ciudades				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	removeCity	setupScenary10	Ninguno	Se ha removido el elemento de la lista, ahora la lista cuenta con tamaño 0, ya no hay elementos, devuelve true.

<b>Objetivo de la prueba:</b> Validar que NO se borre una ciudad de la lista de ciudades				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	removeCity	setupScenary10	Ninguno	No se ha removido la ciudad de la lista, debido a que su referencia es distinta a 0, la lista sigue de tamaño 1 (La ciudad del escenario se le actualiza la ref. a 1)

<b>Objetivo de la prueba:</b> Validar que se agregue correctamente un cliente a las lista de clientes del RentingCar.				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	addClient	setupScenary11	Name = "Gabriel" lastName = "Suarez" Id = "1006325694" refP = "0" codeP = 3 emailC = "Gabriel@gmail.com" phoneC = 3226227443 addressC = "Calle 56"	Se ha creado un cliente en la lista de clientes, este cliente se encuentra en la posición 2 del Array, además de que la lista ahora es de tamaño 2, el cliente fue creado de manera correcta, además de que el email del cliente debe de contener una "@".

<b>Objetivo de la prueba:</b> Validar que NO se agregue un cliente con ID repetido.				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	addClient	setupScenary11	Name = "Gabriel" lastName = "Suarez" Id = "2118442" refP = "0" codeP = 3 emailC = "Gabriel@gmail.com" phoneC = 3226227443 addressC = "56-47"	No se ha agregado el cliente debido a que ya existe un cliente con esa misma Id, se retorna un falso, la lista sigue siendo de tamaño 1.

<b>Objetivo de la prueba:</b> Validar que NO se agregue un cliente sin el "@" en su Email.				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	addClient	setupScenary11	Name = "Gabriel" lastName = "Suarez" Id = "2118442" refP = 0 codeP = 3 emailC = "Gabrielgmail.com" phoneC = 3226227443 addressC = "56-47"	No se ha agregado el cliente debido a que su email esta sin el elemento necesario para que este sea correcto, es decir la arroba.

<b>Objetivo de la prueba:</b> Validar la correcta actualización de un cliente en la lista de clientes.				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	uptadeClient	setupScenary10	Name = "Luis" phoneC = "3226227443"	Se ha actualizado el cliente correctamente, la lista sigue siendo de tamaño 1, y el cliente cuanta con los nuevos parámetros.

<b>Objetivo de la prueba:</b> Validar que NO se actualice un empleado con la misma ID a excepción de el mismo				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	uptadeClient	setupScenary11	(Crear un nuevo cliente) Id = "21118442"	No se ha actualizado el cliente debido a que ya existe otro con la misma ID, sigue con los mimos valores en su atributos.

<b>Objetivo de la prueba:</b> Validar que NO se actualice un empleado sin la "@" en su Email.				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	uptadeClient	setupScenary11	emailC = "Gabrielgmail.com"	No se ha actualizado el cliente debido a que su email no cuenta con el símbolo esperado "@".

<b>Objetivo de la prueba:</b> Validar que se borre un cliente de la lista de clientes				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	removeClient	setupScenary11	Ninguno	Se ha borrado el único elemento de la lista de clientes, ahora la lista cuenta con 0 elementos y es de tamaño 0, además de que las referencias del cliente son igual a 0.

<b>Objetivo de la prueba:</b> Validar que NO se borre un cliente de la lista de clientes debido a que esta referenciado				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	removeClient	setupScenary11		No se ha borrado el cliente de la lista de clientes debido a que esta referenciado en otro objeto, devuelve falso. (Al cliente del escenario se le actualiza la ref. a 1)

<b>Objetivo de la prueba:</b> Validar la correcta creación de una marca.				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	addBrand	setupScenary12	codeB = 2 nameB = "Nissan" refB = 0	Se ha creado una nueva marca, el método devuelve true, además de que la lista de marcas ahora es de tamaño 2, la marca cuenta con sus atributos correctamente implementados.

<b>Objetivo de la prueba:</b> Validar que NO se agregue una marca con el mismo nombre				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	addBrand	setupScenary12	codeB = 2 nameB = "Toyota" refB = 0	No se ha agregado la marca debido a que existe otra con el mismo nombre, el método devuelve false, y la lista sigue siendo de tamaño 1.

<b>Objetivo de la prueba:</b> Validar que se actualice una marca				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	updateBrand	setupScenary12	nameB = "Nissan"	Se ha actualizado correctamente el nombre de la marca, debido a que el nombre no está repetido, devuelve true.

<b>Objetivo de la prueba:</b> Validar que NO se actualice una marca debido a que el nombre esta repetido				
--	--	--	--	--

Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	uptadeBrand	setupScenary12	nameB = "Toyota"	No se ha actualizado correctamente el nombre de la marca, debido a que ya existe otra con el mismo nombre, devuelve false.

**Objetivo de la prueba:** Validar que se borre una marca de la lista de marca

Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	removeBrand	setupScenary12	Ninguno	Se ha removido el elemento de la lista, ahora la lista cuenta con tamaño 0, ya no hay elementos, devuelve true.

**Objetivo de la prueba:** Validar que NO se borre una marca de la lista de marca

Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	removeBrand	setupScenary12	Ninguno	No se ha removido la marca de la lista, debido a que su referencia es distinta a 0, la lista sigue de tamaño 1 (La marca del escenario se le actualiza la ref. a 1)

**Objetivo de la prueba:** Validar la correcta creación de un tipo de vehículo.

Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	addTypeV	setupScenary13	codeTv = 2 nameTv = "4x4" refTv = 0	Se ha creado un nuevo tipo de vehículo, el método devuelve true, además de que la lista de las typeV ahora es de tamaño 2, el tipo de vehículo cuenta con sus atributos correctamente implementados.

**Objetivo de la prueba:** Validar que NO se agregue un tipo de vehículo con el mismo nombre

Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	addTypeV	setupScenary13	codeTv = 2 nameTv = "Deportivo" refTv = 0	No se ha agregado el tipo de vehículo, debido a que existe otro con el mismo nombre, el método devuelve false, y la lista sigue siendo de tamaño 1.

**Objetivo de la prueba:** Validar que se actualice un tipo de vehículo

Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	updateTypeV	setupScenary13	nameTv = "4x4"	Se ha actualizado correctamente el tipo de vehículo, debido a que el nombre no está repetido, devuelve true.

**Objetivo de la prueba:** Validar que NO se actualice un tipo de vehículo debido a que el nombre esta repetido

Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	updateTypeV	setupScenary13	nameTv = "Deportivo"	No se ha actualizado correctamente el tipo de vehículo, debido a que ya existe otro con el mismo nombre, devuelve false.

**Objetivo de la prueba:** Validar que se borre un tipo de vehículo de la lista de typeV

Clase	Método	Escenario	Valores de entrada	Resultado
-------	--------	-----------	--------------------	-----------

RentingCarTest	removeTypeV	setupScenary13	Ninguno	Se ha removido el elemento de la lista, ahora la lista cuenta con tamaño 0, ya no hay elementos, devuelve true.
----------------	-------------	----------------	---------	---

**Objetivo de la prueba:** Validar que NO se borre un tipo de vehículo de la lista de typeV

Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	removeTypeV	setupScenary13	Ninguno	No se ha removido tipo de vehículo de la lista, debido a que su referencia es distinta a 0, la lista sigue de tamaño 1 (El tipo de vehículo del escenario se le actualiza la ref. a 1)

**Objetivo de la prueba:** Validar la correcta creación de un carro.

Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	addCar	setupScenary14	photo = "Null" dispV = true plate = "FXL006" codeV = 6 pricexDay = 80000 model = "A3A" <b>Valores de la marca del carro</b> codeB = 5 name = "Toyota" refB = 1 <b>Valores del tipo del carro</b> codeTv = 4 nameTv = "Deportivo" refTv = 1	Se ha creado el carro correctamente con sus atributos respectivos pasados como parámetros, en la posición siguiente del carro firstV, y el método debe de retornar true.

**Objetivo de la prueba:** Validar que NO se agregue un carro con misma placa

Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	addCar	setupScenary14	photo = "Null" dispV = true plate = "FXL004" codeV = 6 pricexDay = 80000 model = "A3A" <b>Valores de la marca del carro</b> codeB = 5 name = "Toyota" refB = 1 <b>Valores del tipo del carro</b> codeTv = 4 nameTv = "Deportivo" refTv = 1	No se ha agregado el carro, debido a que existe otro con la misma placa, el elemento siguiente de firstV sigue siendo igual a null, devuelve false.

**Objetivo de la prueba:** Validar que se actualice un carro.

Clase	Método	Escenario	Valores de entrada	Resultado
-------	--------	-----------	--------------------	-----------

RentingCarTest	updateCar	setupScenary14	plate = "FXL006" pricexDay = 70000 model = "A24"	Se ha actualizado correctamente el carro, la placa no está repetida, el método devuelve true.
----------------	-----------	----------------	--	---

**Objetivo de la prueba:** Validar que NO se actualice un carro debido a que la placa esta repetida

Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	updateCar	setupScenary14	plate = "FXL004" pricexDay = 70000 model = "A24"	No se ha actualizado correctamente el carro, la placa está repetida, el método devuelve false.

**Objetivo de la prueba:** Validar que se borre un carro de la lista enlazada

Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	removeCar	setupScenary14	Ninguno	Se ha removido el elemento de la lista, ahora el elemento firstV es igual a null, el método devuelve true.

**Objetivo de la prueba:** Validar que NO se borre un vehículo con una referencia mayor a 0

Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	removeCar	setupScenary14	Ninguno	No se ha removido el elemento de la lista, enlazada, el elemento firstV sigue siendo igual a el mismo, el método devuelve false.( El carro del escenario se le actualiza la ref. a 1)

**Objetivo de la prueba:** Validar la correcta creación de una renta.

Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	addRent	setupScenary15	codeR = 12 ticket = A002 fInitial = 27/05/2021 fFinal = 39/05/2021 days = 2 Status = Status Delay = 0 Mult = 0 priceTotal = 1600000 <b>Valores del carro</b> photo = "Null" dispV = true plate = "FXL004" codeV = 9 pricexDay = 80000 model = "A3A" <b>Valores de la marca del carro</b> codeB = 7 name = "Toyota" refB = 1 <b>Valores del tipo del carro</b> codeTv = 8 nameTv = "Deportivo"	Se ha creado una nueva renta, esta renta cuenta con todos sus atributos necesarios, además de que su carro y cliente asociado no son null, la lista ahora tiene tamaño 2, además de que el método devuelve true.

			refTv = 1 <b>Valores del cliente</b> Name = "David" lastName = "Montano" Id = "21118442" refP = "0" codeP = 11 emailC = "David@gmail.com" phoneC = 3127312289 addressC = "Calle 56" <b>Valores de la ciudad del cliente</b> codeCi = 10 nameCi = "Palmira" refCI = 1	
--	--	--	--	--

Objetivo de la prueba: Validar que NO se agregue una renta sin su vehículo				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	addRent	setupScenary15	codeR = 9 ticket = A002 fInitial = 27/05/2021 fFinal = 39/05/2021 days = 2 Status = Status Delay = 0 Mult = 0 priceTotal = 1600000 <b>Valores del Carro = null</b> <b>Valores del cliente</b> Name = "David" lastName = "Montano" Id = "21118442" refP = "0" codeP = 8 emailC = "David@gmail.com" phoneC = 3127312289 addressC = "Calle 56" <b>Valores de la ciudad del cliente</b> codeCi = 7 nameCi = "Palmira" refCI = 1	No se agrego la renta debido a que el carro asociado es igual a null, la lista sigue siendo de tamaño 1, devuelve false.

Objetivo de la prueba: Validar que NO se agregue una renta sin un cliente seleccionado.				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	addRent	setupScenary15	codeR = 10 ticket = A002 fInitial = 27/05/2021 fFinal = 39/05/2021 days = 2	No se agregó la renta debido a que el cliente asociado es igual a null, la lista sigue siendo de tamaño 1, devuelve false.



			Status = Status Delay = 0 Mult = 0 priceTotal = 1600000 <b>Valores del carro</b> photo = "Null" dispV = true plate = "FXL004" codeV = 9 pricexDay = 80000 model = "A3A" <b>Valores de la marca del carro</b> codeB = 7 name = "Toyota" refB = 1 <b>Valores del tipo del carro</b> codeTv = 8 nameTv = "Deportivo" refTv = 1 <b>Valores del cliente</b> = null	
--	--	--	--	--

Objetivo de la prueba: Validar que NO se agregue una renta sin un cliente seleccionado y sin un vehículo				
Clase	Método	Escenario	Valores de entrada	Resultado
RentingCarTest	addRent	setupScenary15	codeR = 7 ticket = A002 fInitial = 27/05/2021 fFinal = 39/05/2021 days = 2 Status = Status Delay = 0 Mult = 0 priceTotal = 1600000 <b>Valores del carro</b> = null <b>Valores del Cliente</b> = null	No se agregó la renta debido a que el cliente y vehículos asociado son iguales a null, la lista sigue siendo de tamaño 1, devuelve false.

