Para comenzar el diseño hicimos una clase User, los cuales tendran un nombre y una contraseña. Se puede acceder al nombre del usuario.

Tanto Admin como Client son usuarios, por lo tanto contaran con un nombre y una contraseña, heredado de su clase padre User.

Un administrador posee 3 listas diferentes:

* clients: posee clientes y podra bloquearlos o desbloquearlos según el uso de los activos.
* purchaseLot: posee los Purchase Lot creadas por el administrador.
* terminal: posee las terminales creadas por el administrador.

Cuales métodos posee:

* blockClient: el cual recibe como parámetro un cliente y su objetivo es bloquear a ese cliente debido a alguna violación en el sistema.
* unblockClient: el cual también recibe como parámetro un cliente y su objetivo es desbloquear a este cliente.
* createApurchaseLot: crea un purchase lot.
* CreateAterminal: permite al administrador crear terminales en diferentes lugares.

Por otro lado creamos una clase Client, modelo de un cliente de la aplicación, este posee un phoneNumber, Money, un Nickname, points, discounts y un estado ya sea de cuenta bloqueada o activa.

Los métodos que poseen los cliente son:

* signUp: tiene como parámetros un PhoneNumber, con este método queremos representar como un usuario se ingresa por primera vez en la aplicación.
* clientPays: tiene como parámetro una Bill y tiene el objetivo de pagar por alquilar cierto activo por un tiempo determinado.
* rentAnActive: tiene como parámetro un Activo, con este método el usuario puede elegir que activó alquilar y alquilarlo.
* howManyPoints: método que muestra cuántos puntos tiene el cliente, este devuelve un int el cual representa la cantidad de puntos que posee ese usuario.
* chooseTimeOfActive: recibe como parámetro un double el cual indica cuánto tiempo va a usar el cliente determinado activo, este método devuelve un double.
* showDiscounts: este método no contiene parámetros y tiene como objetivo mostrar una lista de los descuentos que posee un usuario.

Una de las características de los administradores es crear terminales, para ello creamos una clase llamada Terminal, la cual tiene como variables una lista de activos que posee determinada terminal y los descuentos que se puede obtener en esta.

En cuanto a los métodos son 2:

* givenanActive: retorna un activo y pretende modelar qué pasa cuando un usuario recibe un activo.
* recieveanActive: tiene como parámetro un activo y tiene como objetivo modelar qué ocurre cuando una terminal recibe un activo.

**Active:** Actualmente el activo es la interface que representa los comportamientos de los distintos vehículos. Estos comportamientos son:

* awardPoints: le permite al activo otorgar los puntos al cliente durante la renta del mismo.
* showTariff: este método devuelve el precio que le va a costar al cliente alquilar el activo.
* showDiscount: permite al activo, en caso de que el cliente posea un descuento, mostrar cuál va a ser el precio.
* outOfZoneUse: en el caso de que el cliente se aleje de la zona predeterminada del activo, este método va a retornar una alerta al cliente.

* purchaseLot: el activo posee acceso a esta lista la cual te permite ver los diferentes lotes.
* terminals: esta lista retorna las diferentes terminales en la zona que se encuentra.

**Score:** el score se va a encargar de diferentes tareas relacionadas con los puntos del cliente. Actualmente incluye:

* addClient: este método se encarga de añadir a los clientes a un scoreboard el cual dará recompensas mensualmente a los clientes dentro del top 3 con más puntaje.
* showBestClients: retorna a los clientes dentro del top 10 con más puntaje del scoreboard.
* score: esta lista de clientes va a almacenar al puntaje de cada cliente.
* makeDiscount: este método permite al cliente utilizar sus propios puntos para canjearlos por un descuento en su próximo viaje.

**PurchaseLot:** esta clase contiene una lista de los activos existentes y va a permitir la venta de activos y la creación de los mismos. Contiene los métodos:

* sell: este método retorna un activo vendido.
* createAnActive: permite la creación de nuevos activos.

**System:**

* register: le va a permitir a las personas crear nuevos clientes.

**ABM:**  Esta clase contiene tres métodos los cuales van a permitir tener control sobre los administradores, crearlos, eliminarlos y modificarlos. Estos métodos son:

* createAnAdmin: crea nuevos administradores.
* deleteAnAdmin: permite la eliminación de administradores existentes.
* modifyAnAdmin: modifica administradores existentes.