Giorgio – Méndez – Pérez – Rios - Werner

Una aplicación creada por el Equipo 3 para la gestión de servicios de movilidad, con modo de multi-usuario.

MoovMe

Trabajo Práctico Especial – Introducción a la Programación II



## Gestión de Usuarios:

Nuestro modelo del sistema de usuarios utiliza un árbol de clases a partir del cual, la clase principal y abstracta, sería usuario, mientras que sus clases hijas serian administrador y cliente, y como adicional una clase líder para mostrar en el ranking a los clientes.

Luego para la gestión de los repositorios de cualquier tipo de elemento utilizamos una clase ABM genérica, la cual se encargaría de modificar, agregar y remover. A su vez, el repositorio, se construye sobre una colección del tipo HashSet, esta decisión fue tomada a partir del concepto principal del mismo, el cual impide añadir elementos idénticos.

Por otra parte, para las acciones de registro de usuario y las acciones de los clientes y de los administradores, creamos un menú específico para cada instancia y una clase Screen que se encargaría de implementarlos.

Para el caso de realizar una denuncia como administrador, se puede bloquear un usuario.

## Gestión de activos:

Se crearon las clases para los lotes, activos y tarifarios, implementando en cada una de ellas el comportamiento especificado por la consigna.

En caso de un evento de amenaza de seguridad, se tiene presente en cada cliente si fue bloqueado o no, se termina el viaje en caso de que haya comenzado alguno y se le aplica una multa correspondiente.

Cada lote de compra puede ser creado por un administrador.

## Scoring:

Para el Sistema de scoring creamos las clases scoring y descuento.

Scoring se encargaría de la administración de los descuentos y si se lo piden te devuelve un descuento por zona, tipo de activo y si el cliente cuenta con los puntos requeridos. Además cuenta con una lógica de puntos, la cual se encarga de ordenar los puntos por zona de cada cliente, los cuales se encuentran en un HashMap de Zonas y enteros, dentro de un tablero de líderes mensuales por zonas por medio de un HashMap de zonas y ArrayList de clientes.

Decidimos optar por esta estructura, ya que de esta manera se facilita la clasificación de puntos por cada zona mediante la utilización de las keys por valor, para poder armar correctamente los tableros.

También creamos una clase Líder en la cual almacenaríamos tanto el Alias como el puntaje (en la zona) del cliente, ya que nos facilitaría el almacenamiento de los datos necesarios para el armado del tablero.

Pensamos en poner una bonificación de puntos por zona, para así fomentar el uso de activos en zonas poco frecuentadas.

Por último, Scoring posee un método para premiar a los 3 primeros de cada tablero con un 50% de descuento en su próximo viaje, cambiando el estado Booleano que posee el cliente ganador en True, para así poder canjear el descuento. La entrega del premio posee idempotencia.

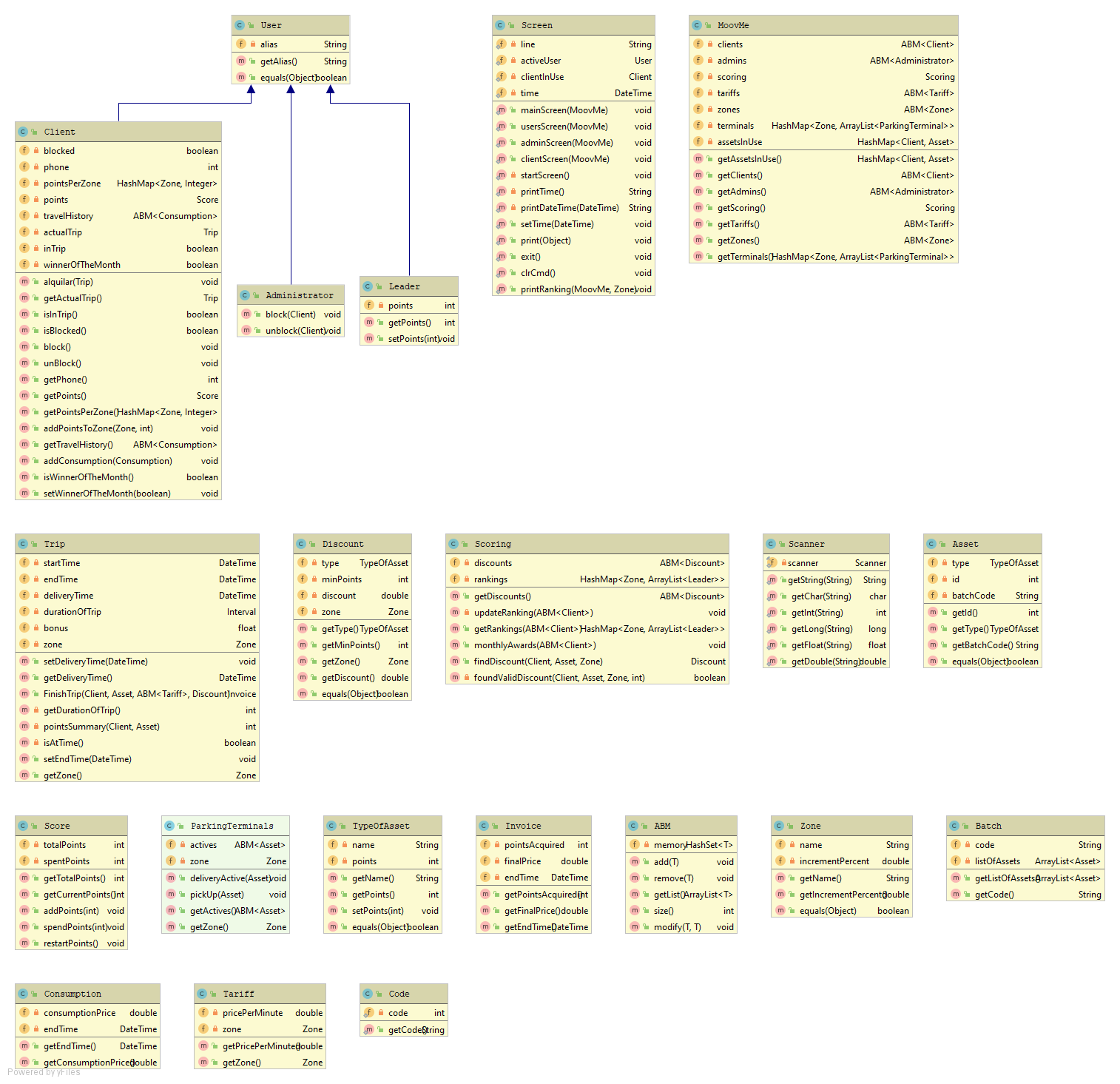
## Pick up y delivery:

En este caso, creamos las clases parkingTerminals en la cual recibimos los activos, y a su vez también los clientes tendrían la posibilidad de devolver el activo en la misma terminal u otra de la misma zona. Para no perder el rastro de los activos, creamos dentro de MoovMe un HashMap activesInUse de activos y clientes, en la cual quedan registrados los activos enlazados a dicho cliente, y en el caso de que suceda un intento de robo, poder rastrear el activo y bloquear al cliente.

Por otra parte, los clientes tendrían la posibilidad de seleccionar un horario de entrega para devolver los activos y así lograr sumar puntos extras, aunque decidimos que si el cliente regresara antes del horario pactado (en caso de que el horario sea coherente) también obtenga ese bonus de puntos. Decidimos hacer uso de Joda Time para el manejo del tiempo.

Pensamos que cada zona debería contar con al menos una terminal por tipo de activo, de esta manera no se juntarían distintos tipos de activos y sería más cómoda la búsqueda por tipo de activo.

## Ejecución de aplicación:

Creamos una clase Screen en la cual por medio de switch cases e if, los usuarios pueden seleccionar entre administradores y clientes, teniendo la posibilidad de ejecutar los métodos mencionados y desarrollados anteriormente.