Tiendas -> Tendrán sus atributos junto a una lista de productos y la cantidad de los mismos
Productos -> Tendrán sus atributos, clase abstracta y usar **patrón de diseño factory** para crearlos
camiones -> **patrón de diseño factory** para crearlos. Cada camión tiene de propiedad el tipo de producto y su cantidad.

Usar **patrón de diseño singleton** para guardar los camiones creados.

Uso de **patrón de diseño mediator** para calcular los precios de las rutas de camiones.

Para la lectura de qr, utilizar **patrón de diseño facade**, donde se usarán funciones de bajo nivel para consumir la api, convertir las clases de la api a qr y viceversa, pero proporcionando un facade donde solo metamos la clase y guardemos el qr, o donde en base al id nos devuelva la clase del qr ya guardado, entre otras funciones que utilizan el api qr o la lectura de las tiendas.

Al dar click en add o modify, se abre otro form pasando por referencia el original. Ya que los datos introducidos sean validados, se crea el elemento con **factory**, se usa el **facade** para qr para guardar el nuevo código, y se llama un **mediator** para actualizar la tabla de valores de los sectores. Esto para ambos botones.

Creational -> builder, **factory**, abstract factory, **singleton**
Behavioral -> bridge, decorator, **facade**, proxy
Structural -> **mediator**, observer, interpreter, command