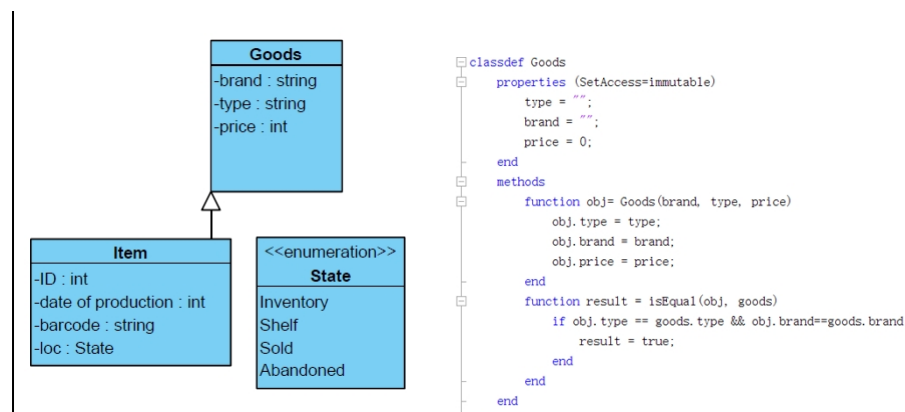
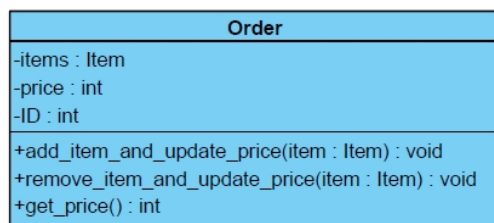


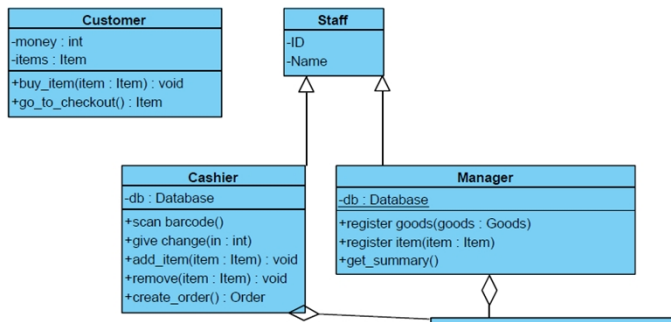
These three System function are implemented successfully. Customer could buy items while database and cashier system dealing with the data of the items. And customer and cashier could generate summary after buying the items. Also, there are some interface for manager to register items.



These two class are specified in src\Goods.m and src\Item.m



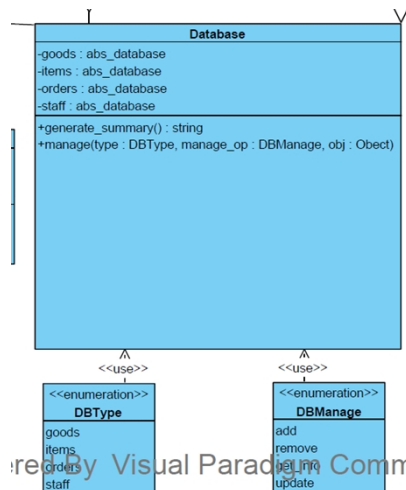
Specified in srd\Order.m



Cashier and Manager are implemented in src\Cashier.m and src\Manager.m

```

----
function obj = create_order(obj)
    obj.order_count = obj.order_count + 1;
    obj.order = Order(obj.order_count);
end
function obj = addItem(obj, item)
    % call the function in Order
    obj.order.add_item_and_update_price(item);
    return;
end
function obj = removeItem(obj, item)
    % call the function in Order
    obj.order.remove_item_and_update_price(item);
    return;
end
function obj = writeOrderBackToDatabase(obj)
    obj.db.writeOrder(obj.order);
    % TODO: save data
    return;
end
  
```



```

classdef Database
    properties
        db_goods;
        db_goods_count;
        db_items;
        db_orders;
        db_orders_count;
        db_staff;
    end
    methods
        function obj = Database()
            if ~exist('data','dir')
                mkdir('data')
            end
            if exist('data/cashier_db.mat','file')
                load 'data/cashier_db.mat' obj
            else
                % init
            end
        end
    end
end
  
```

In src\Database.m. The \*count properties are needed for container.

```

%save 'data/cashier_db.mat' obj
end
function writeData(obj)
    save 'data/cashier_db.mat' obj;
end
function addGoods(obj, brand, type, price)
    obj.db_goods_count = obj.db_goods_count + 1;
    obj.db_goods(obj.db_goods_count) = Goods(brand, type, price);
end
function removeGoods(obj, brand, type, price)
    % TODO
    obj.db_goods_count = obj.db_goods_count + 1;
    obj.db_goods(obj.db_goods_count) = Goods(brand, type, price);
end
function get_InfoGoods(obj)
    % TODO
end
  
```

There are some other function for Database class to write data, which may be used by other class such as Manager or Cashier to operation on database of goods and item.

S1. Customer UI implementation



Customer could buy items through this UI interface. More details are in User Manual.

### S1.1: Buy Goods

#### S1.1.1: Select Goods items

1. Type menus are clicked.
2. Click Goods button in the left to see the detail information of the goods.
3. Click plus button for goods you want, click minus button for goods you do not want to buy.

#### S1.1.2: Pay price

1. Click pay button in the right panel.

#### S1.1.3: Print purchase receipt

1. Click Pay successfully message button in the middle panel.
2. Click print receipt button on the top of the middle panel.

Manager UI is ready

The image shows a software window titled 'UI Figure' with standard Windows window controls (minimize, maximize, close). The window contains a form for registering new goods. At the top, a label reads '您可输入以下信息来登记新的商品：' (You can input the following information to register new goods:). Below this, there are six input fields arranged vertically: '商品品牌名:' (Goods Brand Name), '商品种类:' (Goods Type), '商品货架:' (Goods Shelf) which is a dropdown menu currently showing '美妆洗护' (Beauty Care), '商品价格:' (Goods Price) with a value of '0', '(额外) 商品图片:' (Additional Goods Image), and '商品数量:' (Goods Quantity) with a value of '0'. To the right of these fields is a large button labeled '登记' (Register).

## S2. Manager UI implementation

### S2.1.1: Register Goods

#### S2.1.1.1: input information of goods

1. Input goods name
2. Input goods type.
3. Input the remain goods information.

#### S2.1.1.2: check validation of information

1. Check the validation of the information you have input, or you will receive error message.

#### S2.1.1.3: register

1. Click register button.