Code Document

Name: Li Kaixu

Student Id: 1155180259

When you deploy the contract, the IO and functions of API functions are listed below:

1. **BuyerRegistration(string name, string email, string shippingAddress)**

Registration of a new buyer. The input can be empty string.

1. **SellerRegistration(string name)**

Seller registration. You only need to input a name. The first valid address with at least 1 ether msg.value will be choosen.

1. **BuyerProfileUpdate(string name, string email, string shippingAddress)**

Update of buyer information. The sender address needs to be registered as a buyer before. We do not allow seller to change his information.

1. **PutOnProduct(uint id, string name, uint price, uint quantity)**

Seller put on a new product on shopping mall. This function simply change product[id] = {id, name, price, quantity}. So it will simply cover the information of product if the id is used before.

We punish the seller by 1 ether if the price > 20 ether. We send punishment from this.address to seller address.

1. **PrintProductInformation(uint id)**

Print the information of product[id], anyone can see it, event the unregistered ones.

1. **TransactionInitiation(uint product\_id,uint product\_number)**

Only valid buyers can initiation a transaction to buyer certain number of a product. At least buy 1 and he need to have enough msg.value . The seller should have enough amount of product or it will fail.

1. **ReturnRequestMyId(uint my\_transaction\_id)**

The buyer of a particular transaction can ask for a return before completion. Pay attention this is transaction id is in buyer’s view. The index is 0,1,2,… rather than real id.

1. **AllowReturnRequest(uint transaction\_id)**

Only the seller can allow the return request after the buyer send a return request. After that the payment of this transaction will be sent to the buyer. The input is real id in chain.

1. **TransactionCompletion(uint my\_transaction\_id)**

The buyer of a transaction can set it as completed. If so, nothing should be changed about this transaction. Transaction id is in buyer’s view. The index is 0, 1, 2, … rather than real id.

1. **ChangeScore(uint my\_transaction\_id,uint my\_score)**

The user can rate the score of a transaction from 0-5. The default score is 5. If you input a score > 5, it will automatically change to 5. The id is still in buyer view

1. **PrintTransaction()**

Print the list of transaction:

Seller: see all the transaction information

Buyer: only see his transaction

1. **PrintPersonalInfo()**

We allow the buyers to view its personal information, including the real transaction id without details.

1. **Donate()**

We allow all address to transfer ether to the contract address to maintain its regular use. You change the input msg.value(at least 1 ether) and the balance will receive it. In my view only seller will have the motion but we do no restriction.

Some public variables:

These variables have no relation to the regular use of this contract. They exists mostly because I am not willing or too lazy to set it as private.

1. basic\_price: It is set 1 ether for better view. All input and output price/payment is in uint of 1 ether. You can change it but ether is most obvious.
2. my\_seller: the information of seller, including name, address, seller\_in
3. product\_map: the mapping of product\_id => product info. This is the same if you use PrintProductInfo. I think it is ok.
4. register\_arr/register\_list: the arr is the list of address of seller and buyer. The list is the mapping of address=>bool to check if an address is valid. This 2 exists most because I am not willing to set it private. I think it is ok since you can not see detailed description, only the addresses.