Project instructions sheet

in our code there is three main packages, (interface layer, presentation layer, business layer).

Starting with the **presentation layer** witch has one class called mainMenu , this class holds a field of type systemManager witch calls all the functions of the project , the purpose of This class is to handle all the inputs from the user and according to his choice it sends the input to the correct function, (it allows him to do things in the super).

The **interface layer** has one class called systemManager, this class holds two major fields (business controller, user controller) in order to call the right function according to the user, all the functions of the system is written in here in order to call the same function from the bottom tier the business layer. For example, the user wants to sign in this function only passes the inputs to the same function in the business layer where all the logical work is done.

The **business layer** contains all the DTO's and the two main controllers (business, user) the main object of this tier is the:

- **user** class: holds the name email and password for every user and has setters and getters functions.

- **report** class: holds the date of the report, and its title, in addition to setters/getters.

- **item** class: holds the item name , expiration date, a Boolean value for defect ,setters getters.

- **discount** class: holds the starting and end date of the discount, and its percentage, and a unique ID, in addition to setters and getters.

- **itemdiscount** class: implements discount class holds one fields of type itemSpecs that the discount is applied on,
and two main functions:
(**withDiscount**) returns the percentage and the expiration date as string
(**validItemDiscount**) returns false if the discount has expired for this item

Categorydiscount class: implements discount class and holds one field of type category that the discount is applied on,
and two main functions:

(**validCategoryDiscount**) returns false if the discount has expired for this category
(**withDiscount**) returns the percentage and the expiration date as string.

UserController class: it has one map and two linkedlists to store the required data, it has two functions one that checks the validation of the email and one for the password, and a login / logout functions in addition to a register function that checks the user details .

itemSpecs class: this class is built in order the manage items of the same type but with different expiration date, we store in this class the items total amount, shelf amount, stored amount, minimum amount, and the price with/without discounts, every itemSpecs class have a linkedlist for all the items of this type , they are the almost the same object but the only difference is the expiration date of each item and the defect field in each one , one item must be defected but the other is not and they are both of the same kind.
There is a set price function that changes the price to the given one,
set discount: that changes the discount and updates the new price,
get amounts: to get all the information about the amounts in shelf storage and in total, setters/getters for all the amounts (total, stored, minimum..) ,
remove item: function that searches for the specific item ID and deletes it from the list,
remove from shelf: reduces the shelf amount by a given amount.
move to shelf: increases the shelf amount ,
add item: adds a new item to the list and increases the storage and total amount by one.

BusinessController class: this class has all the main functions , it has one list and two maps to store the data with two counters for the item and category starting from 1,
the functions in this class is called from the tier above almost all of them only receives one parameter in the input which is serialNumber of type String, the serialNumber is a unique string that is built from numbers the first three is the category id, second three is the sub category id, and the third thee is the subsub category id , in addition to six digits at the end which are the item id .
all the functions in this class was built also in the category class the only difference that in the business controller class we search for the wanted category and when we find it we just calls the same function but from the category class (in other words we want to work inside the category so we find it first and then calls the function that is inside the category class to let it work from the inside) this help us to keep the code clean and readable.
Im going to explain the purpose of each function in the category class, because what these functions do here is searching for the right category and then passes the values to the inside function.

category class: contains one map to save the sub categories and a list for the itemSpecs , a level of type int , a name and ID.
main functions:
Add item: it adds the item to the list by creating a new itemSpecs and check whether it has a discount or not and adds it.
add sub category: adds a new category to the map of the current category.
check amount: checks if the amount of specific item is >= of a given amount in this category.

remove item: deletes the item from the list.
check serial number: checks if this serial number really exist.
transfer item: it moves the items from storage to shelfs .
make min items report: it returns a string to be printed , that contain all the items that their total amount have reached the minimum amount.
make lack report: it returns a string to be printed , that contain all the items that their total amount is less than the minimum amount.
make defect report: it returns a string to be printed , that contain all the items that has been defected (their defect field is true)
making item report: it searches for the right item and returns a string to be printed that contains all the information of this specific item.
check category name legality: returns false if the name is already used.
making category report: it returns a string to be printed , that contain all the items that is in this category.
adding defect item: searches for the item and changes its defect field to true.
add item discount: searches for the item and add to is the new discount.
add cat discount: sets a new discount for all the items in this category .
in addition to basic setters/ getter.