GTU Department of Computer Engineering
CSE 222/505 - Spring 2021
Homework 3 Report

Name : Md Sarwar Hossain
Student Num : 161044121

Part One

1. SYSTEM REQUIREMENTS

The Furniture Shop automatic system includes some branches and with specific kind of products. There will be three types of user (customer, branch employee, administrator). All three user has specific kind of task they can perform.

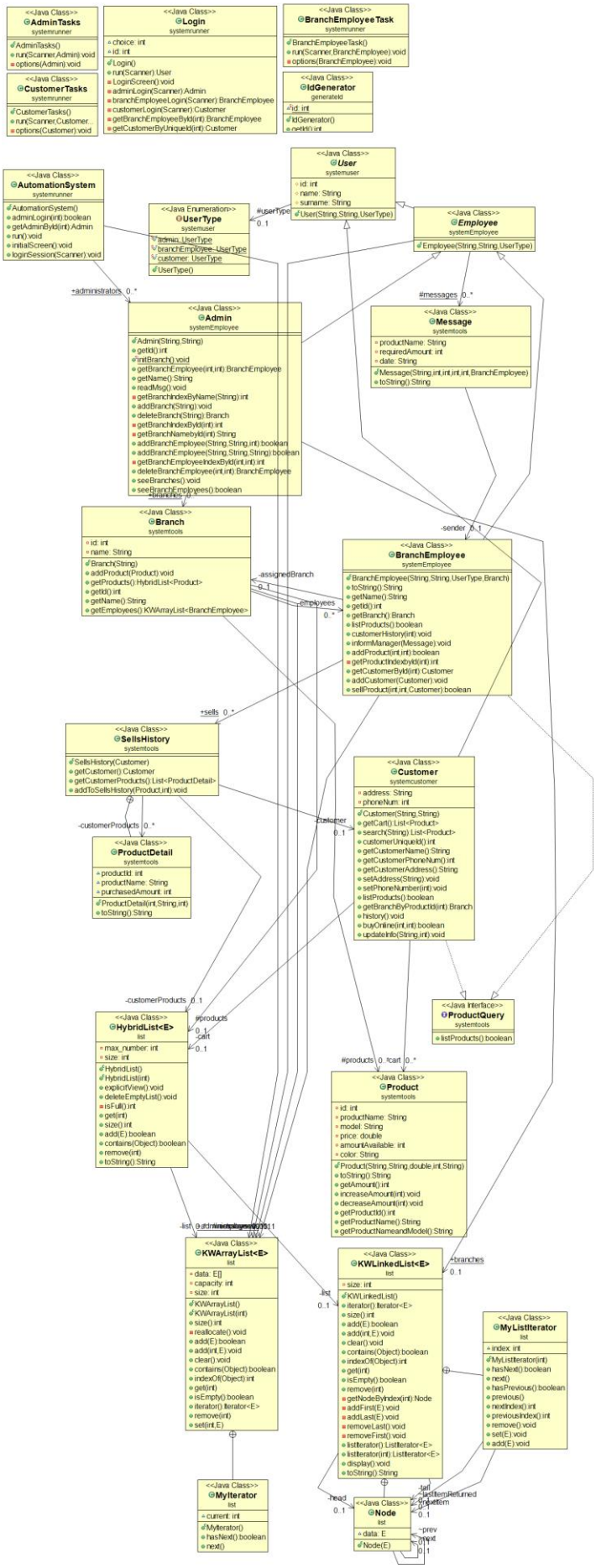
Admin can add / remove branch or branch employee and check messages about product sent by branch employee. Admin can also keep track of all branch employees at once.

Branch employee can add / remove product (mainly increase or decrease product quantity), can inform admin panel if they need any product in a branch if products run out of stock. Branch employee also sell product to a customer. They can also inquire about a customer's purchase history.

Customer interact with the system by searching for a product or look at the all products. They can also look in which branch a product exist. Customer should have the capability to buy product online and keep track of his/her purchase history.

These are the main things the system should have to perform all the task automatically. To resolve all these problem I tried to follow a systematic way explained below.

2. Class Diagram


<<Java Class>>
AdminTasks
systemrunner
AdminTasks()
run(Scanner,Admin):void
options(Admin):void
<<Java Class>>
CustomerTasks
systemrunner
CustomerTasks()
run(Scanner,Customer)
options(Customer):void
<<Java Class>>
Login
systemrunner
choice: int
id: int
Login()
run(Scanner):User
LoginScreen():void
adminLogin(Scanner):Admin
branchEmployeeLogin(Scanner):BranchEmployee
customerLogin(Scanner):Customer
getBranchEmployeeById(int):BranchEmployee
getCustomerByUniqueId(int):Customer
<<Java Class>>
BranchEmployeeTask
systemrunner
BranchEmployeeTask()
run(Scanner,BranchEmployee):void
options(BranchEmployee):void
<<Java Class>>
IdGenerator
generateId
id: int
IdGenerator()
getId():int
<<Java Class>>
AutomationSystem
systemrunner
AutomationSystem()
adminLogin(int):boolean
getAdminById(int):Admin
run():void
initialScreen():void
loginSession(Scanner):void
<<Java Class>>
User
systemuser
id: int
name: String
surname: String
User(String,String,UserType)
<<Java Enumeration>>
UserType
systemuser
admin: UserType
branchEmployee: UserType
customer: UserType
UserType()
<<Java Class>>
Employee
systemEmployee
Employee(String,String,UserType)
<<Java Class>>
Admin
systemEmployee
Admin(String,String)
getId():int
initBranch():void
getBranchEmployee(int,int):BranchEmployee
getName():String
readMsg():void
getBranchIndexByName(String):int
addBranch(String):void
deleteBranch(String):Branch
getBranchIndexById(int):int
getBranchNameById(int):String
addBranchEmployee(String,String,int):boolean
addBranchEmployee(String,String,String):boolean
getBranchEmployeeIndexById(int,int):int
deleteBranchEmployee(int,int):BranchEmployee
seeBranches():void
seeBranchEmployees():boolean
<<Java Class>>
Message
systemtools
productName: String
requiredAmount: int
date: String
Message(String,int,int,int,int,BranchEmployee)
toString():String
<<Java Class>>
Branch
systemtools
id: int
name: String
Branch(String)
addProduct(Product):void
getProducts():HybridList<Product>
getId():int
getName():String
getEmployees():KWArrayList<BranchEmployee>
<<Java Class>>
BranchEmployee
systemEmployee
BranchEmployee(String,String,UserType,Branch)
toString():String
getName():String
getId():int
getBranch():Branch
listProducts():boolean
customerHistory(int):void
informManager(Message):void
addProduct(int,int):boolean
getProductIndexById(int):int
getCustomerById(int):Customer
addCustomer(Customer):void
sellProduct(int,int,Customer):boolean
<<Java Class>>
SellsHistory
systemtools
SellsHistory(Customer)
getCustomer():Customer
getCustomerProducts():List<ProductDetail>
addToSellsHistory(Product,int):void
<<Java Class>>
Customer
systemcustomer
address: String
phoneNum: int
Customer(String,String)
getCart():List<Product>
search(String):List<Product>
customerUniqueId():int
getCustomerName():String
getCustomerPhoneNum():int
getCustomerAddress():String
setAddress(String):void
setPhoneNumber(int):void
listProducts():boolean
getBranchByProductId(int):Branch
history():void
buyOnline(int,int):boolean
updateInfo(String,int):void
<<Java Class>>
ProductDetail
systemtools
productId: int
productName: String
purchasedAmount: int
ProductDetail(int,String,int)
toString():String
<<Java Interface>>
ProductQuery
systemtools
listProducts():boolean
<<Java Class>>
HybridList<E>
list
max_number: int
size: int
HybridList()
HybridList(int)
explicitView():void
deleteEmptyList():void
isFull():int
get(int)
size():int
add(E):boolean
contains(Object):boolean
remove(int)
toString():String
<<Java Class>>
Product
systemtools
id: int
productName: String
model: String
price: double
amountAvailable: int
color: String
Product(String,String,double,int,String)
toString():String
getAmount():int
increaseAmount(int):void
decreaseAmount(int):void
getProductId():int
getProductName():String
getProductNameandModel():String
<<Java Class>>
KWArrayList<E>
list
data: E[]
capacity: int
size: int
KWArrayList()
KWArrayList(int)
size():int
reallocate():void
add(E):boolean
add(int,E):void
clear():void
contains(Object):boolean
indexOf(Object):int
get(int)
isEmpty():boolean
iterator():Iterator<E>
remove(int)
set(int,E)
<<Java Class>>
KWLinkedList<E>
list
size: int
KWLinkedList()
iterator():Iterator<E>
size():int
add(E):boolean
add(int,E):void
clear():void
contains(Object):boolean
indexOf(Object):int
get(int)
isEmpty():boolean
remove(int)
getNodeByIndex(int):Node
addFirst(E):void
addLast(E):void
removeLast():void
removeFirst():void
listIterator():ListIterator<E>
listIterator(int):ListIterator<E>
display():void
toString():String
<<Java Class>>
MyListIterator
list
index: int
MyListIterator(int)
hasNext():boolean
next()
hasPrevious():boolean
previous()
nextIndex():int
previousIndex():int
remove():void
set(E):void
add(E):void
<<Java Class>>
MyIterator
list
current: int
MyIterator()
hasNext():boolean
next()
<<Java Class>>
Node
list
data: E
Node(E)
+administrators 0..*
#userType 0..1
#messages 0..*
+branches 0..*
-sender 0..1
-assignedBranch 0..1
-employees 0..*
+sells 0..*
-customerProducts 0..*
-customer 0..1
-customerProducts 0..1
#products 0..1
-cart 0..1
#products 0..*
-cart 0..*
-list 0..1
+branches 0..1
-list 0..1
-head 0..1
-tail 0..1
-lastItemReturned 0..1
-nextItem 0..1
~prev 0..1
~next 0..1

2.

Use Case Diagram :

3. PROBLEM SOLUTION APPROACH

For the solution of this problem I used a specific design pattern . There are different models for arranging information and to keep this information in a pattern I used my own list architecture. There are KWArrayList, KWLinkedList and HybridList which I used to keep data.

To hold branch information I have Branch class. There are there different classes for three kinds of user. There is product class to hold information about the product. This is the main backbone of the automation system.

Besides to help those classes I have some extra classes e.g: Message class to make sure message between administrator and employees are handled in a formal way. There is a IdGenerator class to create some random id. SellHistory class to keep track of sales in a more maintainable manner .

Mostly the system needs administrator to perform some specific task , same for customers and branch employee. To layout their task gracefully I added those option in systemrunner package. Ultimately using this system the shopping system will run automatically.

As humans we can make error. So I tried to make all possible error a system user can make and inform the user about their error. The system can be run by a user without any prior knowledge. For that I added all the options clearly with instructions.

4. TEST CASES

Case 1 : Register as Admin

Case 2 : Login as Admin with correct credential

Case 3 : Login as Admin with Invalid credential

Case 4: Add Branch

Case 5: Add Branch with same name as existing Branch (duplicate name not allowed)

Case 6: Remove Branch

Case 7: See Branches

Case 8: Add Branch Employee and assign a branch to employee

Case 9: Add Branch Employee and assign to an invalid branch

Case 10: Remove Branch Employee

Case 11: See Branch Employee

Case 12: check messages sent by branch employees

Case 13: Login as Branch Employee with valid credential

Case 14: Login as Branch Employee with invalid credential

Case 15: See Products of a Branch

Case 16: Inform manager about shortage of a product

Case 17: Add product(increase a product amount) with valid product id

Case 18: Add product by providing invalid product id

Case 19: Sell Product to a new Customer

Case 20: Sell Product to an existing customer

Case 21: Sell Product when product is in stock

Case 22: Sell a product when product is not enough in stock

Case 23: Login as Customer with valid credential

Case 24: Search Product by Product Name (case sensitive)

Case 25: Search Product by a product name which is not in store

Case 26: List of all product in the system

Case 27 : Find a product’s branch by product id

Case 28: Shop Online

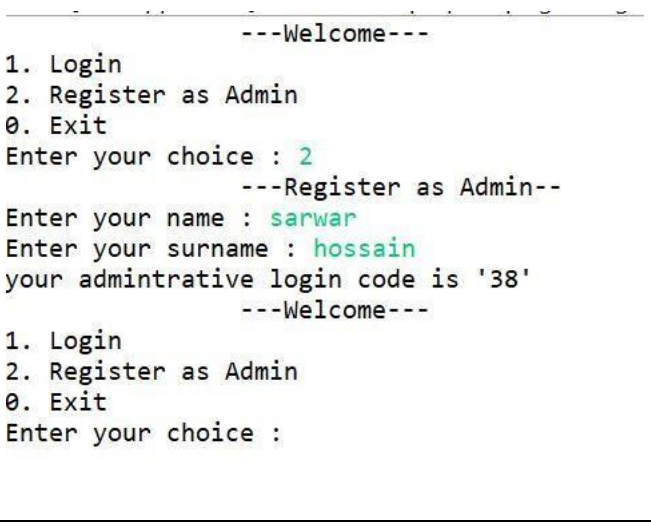
Case 29: Shop online when product is not enough in the store

Case 30: Shopping history of a Customer

Case 31: Update Billing address for online buying

Case 32: Invalid input check when taking input from user

5. RUNNING AND RESULTS

Test Case	Expected Output	Result	Screenshot
Case 1	Admin should register to system and get a code to login to system	pass	

Case 2	Show welcome screen to admin user	Pass	---Login--- 1. Login as Admin 2. Login as Branch Employee 3. Login as Customer 0. Exit Enter your choice : 1 ---Admin Login--- Please Enter Your Secret Login Code : 38 ---Welcome sarwar--- 1. Add Branch 2. Remove Branch 3. See Branches 4. Add Branch Employee 5. Remove Branch Employee 6. See Branch Employees 7. Check Messages 0. Logout Enter your choice :
Case 3	Print invalid credential as output	Pass	

			---Login--- 1. Login as Admin 2. Login as Branch Employee 3. Login as Customer 0. Exit Enter your choice : 1 ---Admin Login--- Please Enter Your Secret Login Code : 40 Invalid Credential !!!
Case 4	Branch added message	pass	1. Add Branch 2. Remove Branch 3. See Branches 4. Add Branch Employee 5. Remove Branch Employee 6. See Branch Employees 7. Check Messages 0. Logout Enter your choice : 1 Please Enter the Branch Name : Rajshahi Rajshahi added as new branch

Case 5	Branch already exists warning message	Pass	1. Add Branch 2. Remove Branch 3. See Branches 4. Add Branch Employee 5. Remove Branch Employee 6. See Branch Employees 7. Check Messages 0. Logout Enter your choice : 1 Please Enter the Branch Name : Birampur Branch already exists !!!

Case 6	Branch removed message	Pass	1. Add Branch 2. Remove Branch 3. See Branches 4. Add Branch Employee 5. Remove Branch Employee 6. See Branch Employees 7. Check Messages 0. Logout Enter your choice : 2 ---List of Branches--- Branch Id : 1 Branch Name : Birampur Branch Id : 2 Branch Name : Dinajpur Branch Id : 3 Branch Name : Rangpur Branch Id : 4 Branch Name : Dhaka Branch Id : 39 Branch Name : Rajshahi Please Enter the Branch Name to delete : Birampur Birampur branch is removed

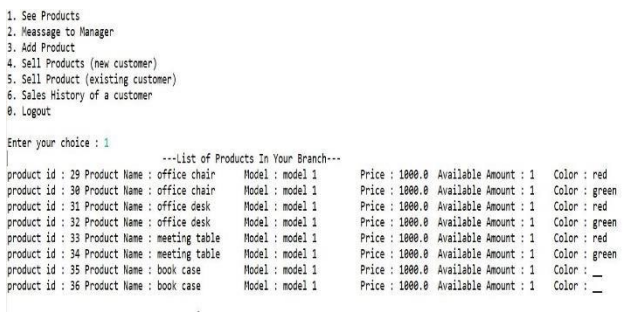
Case 7	Show all the branches in the system	Pass	---Welcome sarwar--- 1. Add Branch 2. Remove Branch 3. See Branches 4. Add Branch Employee 5. Remove Branch Employee 6. See Branch Employees 7. Check Messages 0. Logout Enter your choice : 3 ---List of Branches--- Branch Id : 1 Branch Name : Birampur Branch Id : 2 Branch Name : Dinajpur Branch Id : 3 Branch Name : Rangpur Branch Id : 4 Branch Name : Dhaka
Case 8	Adding a branch employee to system and give the employee a unique id to login into the system	Pass	

			1. Add Branch 2. Remove Branch 3. See Branches 4. Add Branch Employee 5. Remove Branch Employee 6. See Branch Employees 7. Check Messages 0. Logout Enter your choice : 4 Please Enter Employee Name : Abdullah Please Enter Employee Surname: Sakib ---List of Branches--- Branch Id : 2 Branch Name : Dinajpur Branch Id : 3 Branch Name : Rangpur Branch Id : 4 Branch Name : Dhaka Branch Id : 39 Branch Name : Rajshahi Branch Id : 40 Branch Name : rajshahi Assign a branch to this employee(enter name/id of the branch) : 2 Employee Id : 41 (need this id to login later)

Case 9	Trying to assign a invalid branch to a branch employee will cause error	Pass	---Branch Employees--- No Branch employees!!!. Try adding them first ---Welcome abdullah--- 1. Add Branch 2. Remove Branch 3. See Branches 4. Add Branch Employee 5. Remove Branch Employee 6. See Branch Employees 7. Check Messages 0. Logout Enter your choice : 4 Please Enter Employee Name : abdullah Please Enter Employee Surname: sakib ---List of Branches--- Branch Id : 1 Branch Name : Birampur Branch Id : 2 Branch Name : Dinajpur Branch Id : 3 Branch Name : Rangpur Branch Id : 4 Branch Name : Dhaka Assign a branch to this employee(enter name/id of the branch) : gebze Invalid Branch!!!

Case 10	Should print a employee removed message	Pass	5. Remove Branch Employee 6. See Branch Employees 7. Check Messages 0. Logout Enter your choice : 5 ---Branch Employees--- Employee Id : 39 Name : abdullah sakib UserType : branchEmployee branchId : 1 branchName : Birampur Please Enter 'Id' of employee you want to delete : 39 Please Enter Employee's Branch Id : 1 abdullah sakib is removed !!!

Case 11	Should print all the branch employees	pass	1. Add Branch 2. Remove Branch 3. See Branches 4. Add Branch Employee 5. Remove Branch Employee 6. See Branch Employees 7. Check Messages 0. Logout Enter your choice : 6 ---Branch Employees--- Employee Id : 44 Name : Sarwar Hossain UserType : branchEmployee branchId : 4 branchName : Dhaka
Case 12	Should read all the message sent by branch employee	Pass	1. Add Branch 2. Remove Branch 3. See Branches 4. Add Branch Employee 5. Remove Branch Employee 6. See Branch Employees 7. Check Messages 0. Logout Enter your choice : 7 1 : We need 50 Book in Birampur branch posted by : 6 4 (id : 41) date : 5/5/2021 2 : We need 50 Book in Dhaka branch posted by : sarwar hossain (id : 42) date : 2/2/2021

Case 13	should welcome branch employee	Pass	1. Login as Admin 2. Login as Branch Employee 3. Login as Customer 0. Exit Enter your choice : 2 ---Branch Employee Login--- Please Enter Your Employee Id : 45 ---Welcome Sarwar Hossain--- 1. See Products 2. Meassage to Manager 3. Add Product 4. Sell Products (new customer) 5. Sell Product (existing customer) 6. Sales History of a customer 0. Logout Enter your choice :
Case 14	Should not allow to login as branch employee	Pass	1. Login as Admin 2. Login as Branch Employee 3. Login as Customer 0. Exit Enter your choice : 2 ---Branch Employee Login--- Please Enter Your Employee Id : 42 Invalid Credential !!!
Case 15	Should print all the products in the branch	Pass	
Case 16	Should compose a message and send to administrators	Pass	

			---Branch Employee Login--- Please Enter Your Employee Id : 39 ---Welcome sarwar hossain--- 1. See Products 2. Meassage to Manager 3. Add Product 4. Sell Products (new customer) 5. Sell Product (existing customer) 6. Sales History of a customer 0. Logout Enter your choice : 2 Please name the product you need : office chair Please enter the required amount : 10 Please enter today's day (0 - 30) : 13 Please enter today's month (0 - 12) : 3 Please enter year (e.g:2021) : 2021 ---Welcome sarwar hossain---
Case 17	Should increase the amount of product and display product added message	Pass	1. See Products 2. Meassage to Manager 3. Add Product 4. Sell Products (new customer) 5. Sell Product (existing customer) 6. Sales History of a customer 0. Logout Enter your choice : 3 ---List of Products In Your Branch--- product id : 29 Product Name : office chair Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 30 Product Name : office chair Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 31 Product Name : office desk Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 32 Product Name : office desk Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 33 Product Name : meeting table Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 34 Product Name : meeting table Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 35 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ product id : 36 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ Please enter the product id : 29 Enter amount you want to add : 50 Products added
Case 18	İf product id is invalid then should show a message saying failed to add product	Pass	1. See Products 2. Meassage to Manager 3. Add Product 4. Sell Products (new customer) 5. Sell Product (existing customer) 6. Sales History of a customer 0. Logout Enter your choice : 3 ---List of Products In Your Branch--- product id : 29 Product Name : office chair Model : model 1 Price : 1000.0 Available Amount : 51 Color : red product id : 30 Product Name : office chair Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 31 Product Name : office desk Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 32 Product Name : office desk Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 33 Product Name : meeting table Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 34 Product Name : meeting table Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 35 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ product id : 36 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ Please enter the product id : 28 Enter amount you want to add : 1 Failed to add Product !!!

Case 19	Should sell product to a new customer and print product sold	Pass	

	message		1. See Products 2. Meassage to Manager 3. Add Product 4. Sell Products (new customer) 5. Sell Product (existing customer) 6. Sales History of a customer 0. Logout Enter your choice : 4 Please Enter Customer Name : Abdullah Please Enter Customer Surname : Sakib ---List of Products In Your Branch--- product id : 29 Product Name : office chair Model : model 1 Price : 1000.0 Available Amount : 51 Color : red product id : 30 Product Name : office chair Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 31 Product Name : office desk Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 32 Product Name : office desk Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 33 Product Name : meeting table Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 34 Product Name : meeting table Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 35 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ product id : 36 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ Please enter the product id : 29 Enter amount : 1 Product Sold !!! Customer's Unique Id : 47. Customer will use this ID to login in the system
Case 20	Should sell product to a existing customer if provide customer id	Pass	1. See Products 2. Meassage to Manager 3. Add Product 4. Sell Products (new customer) 5. Sell Product (existing customer) 6. Sales History of a customer 0. Logout Enter your choice : 5 Please Enter Customer's Unique Id : 47 ---List of Products In Your Branch--- product id : 29 Product Name : office chair Model : model 1 Price : 1000.0 Available Amount : 50 Color : red product id : 30 Product Name : office chair Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 31 Product Name : office desk Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 32 Product Name : office desk Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 33 Product Name : meeting table Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 34 Product Name : meeting table Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 35 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ product id : 36 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ Please enter the product id : 29 Enter amount : 20 Product Sold !!!

Case 21	Shouldn't sell a product if not enough in stock	Pass	1. See Products 2. Meassage to Manager 3. Add Product 4. Sell Products (new customer) 5. Sell Product (existing customer) 6. Sales History of a customer 0. Logout Enter your choice : 5 Please Enter Customer's Unique Id : 47 ---List of Products In Your Branch--- product id : 29 Product Name : office chair Model : model 1 Price : 1000.0 Available Amount : 30 Color : red product id : 30 Product Name : office chair Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 31 Product Name : office desk Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 32 Product Name : office desk Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 33 Product Name : meeting table Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 34 Product Name : meeting table Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 35 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ product id : 36 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ Please enter the product id : 29 Enter amount : 40 Not enough product in stock !!!
Case 22	Should show customers purchase history	Pass	

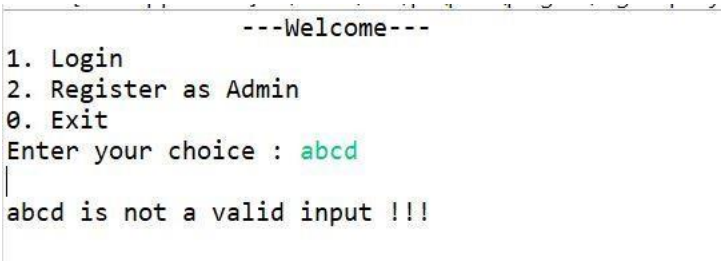
			1. See Products 2. Meassage to Manager 3. Add Product 4. Sell Products (new customer) 5. Sell Product (existing customer) 6. Sales History of a customer 0. Logout Enter your choice : 6 Please Enter Customer's Unique Id : 47 ---Customer History--- customer id : 47 name : Abdullah Product id : 29 Product : office chair model : model 1 purchased amount : 21

Case 23	İf valid id is provided then should show welcome screen to customer	Pass	1. Login as Admin 2. Login as Branch Employee 3. Login as Customer 0. Exit Enter your choice : 3 ---Customer Login--- Please Enter Your Unique Id : 47 ---Welcome Abdullah--- 1. Search Product 2. List of Products 3. Find a Product's Branch 4. Shop Online 5. Shopping History 6. Update billing address 0. Logout Enter your choice :
Case 24	Should print list of product in different branches if searched by product name	Pass	1. Search Product 2. List of Products 3. Find a Product's Branch 4. Shop Online 5. Shopping History 6. Update billing address 0. Logout Enter your choice : 1 Please Enter a Product Name : book case ---Products by Search Query--- product id : 19 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ product id : 20 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ product id : 27 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ product id : 28 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ product id : 35 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ product id : 36 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _
Case 25	Should show warning message if product is not found	Pass	

			---Customer Login--- Please Enter Your Unique Id : 41 ---Welcome setara--- 1. Search Product 2. List of Products 3. Find a Product's Branch 4. Shop Online 5. Shopping History 6. Update billing address 0. Logout Enter your choice : 1 Please Enter a Product Name : something weird No Product Found !!!
Case 26	Should print all the products in the system	Pass	1. Search Product 2. List of Products 3. Find a Product's Branch 4. Shop Online 5. Shopping History 6. Update billing address 0. Logout Enter your choice : 2 ---List of All Products--- product id : 13 Product Name : office chair Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 14 Product Name : office chair Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 15 Product Name : office desk Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 16 Product Name : office desk Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 17 Product Name : meeting table Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 18 Product Name : meeting table Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 19 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ product id : 20 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ product id : 21 Product Name : office chair Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 22 Product Name : office chair Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 23 Product Name : office desk Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 24 Product Name : office desk Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 25 Product Name : meeting table Model : model 1 Price : 1000.0 Available Amount : 1 Color : red product id : 26 Product Name : meeting table Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 27 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ product id : 28 Product Name : book case Model : model 1 Price : 1000.0 Available Amount : 1 Color : _ product id : 29 Product Name : office chair Model : model 1 Price : 1000.0 Available Amount : 30 Color : red product id : 30 Product Name : office chair Model : model 1 Price : 1000.0 Available Amount : 1 Color : green product id : 31 Product Name : office desk Model : model 1 Price : 1000.0 Available Amount : 1 Color : red
Case 27	Should find a product's branch if valid product id is provided	Pass	1. Search Product 2. List of Products 3. Find a Product's Branch 4. Shop Online 5. Shopping History 6. Update billing address 0. Logout Enter your choice : 3 Please Enter a Product id : 13 ---Product Available in following branch--- Branch Id : 2 Branch Name : Dinajpur

Case 28	For shop online if the billing address doesn't exist	pass	

	then will ask for it and then customer will be able to buy product if available in stock		1. Search Product 2. List of Products 3. Find a Product's Branch 4. Shop Online 5. Shopping History 6. Update billing address 0. Logout Enter your choice : 4 Enter your current Address : Birampur Enter your mobile number : 01715412057 Enter Product Id : 19 Enter Amount : 1 Product Purchased Successfully !!!
Case 29	İf product is not sufficient of invalid product id is given then should print couldn't buy product	Pass	1. Search Product 2. List of Products 3. Find a Product's Branch 4. Shop Online 5. Shopping History 6. Update billing address 0. Logout Enter your choice : 4 ---Your current billing address and phone number--- Address : Birampur phone number : 1715412057 Enter Product Id : 19 Enter Amount : 50 Couldn't buy Product !!!

Case 30	Should print a customers purchase history	Pass	1. Search Product 2. List of Products 3. Find a Product's Branch 4. Shop Online 5. Shopping History 6. Update billing address 0. Logout Enter your choice : 5 ---Your Shopping History--- customer id : 44 name : Sarwar Product id : 29 Product : office chair model : model 1 purchased amount : 1 Product id : 19 Product : book case model : model 1 purchased amount : 1
Case 31	Customer should be able to	Pass	
	update his/her billing address		1. Search Product 2. List of Products 3. Find a Product's Branch 4. Shop Online 5. Shopping History 6. Update billing address 0. Logout Enter your choice : 6 ---Your current billing address and phone number--- Address : Birampur phone number : 1715412057 Enter your current address : Dhaka Enter your phone number : 01715412057 Updated !!!
Case 32	For any kind of input mismatch will inform user about it by showing a warning message	pass	

Part Two

Time Complexity Analysis:

Administrator :

Branches are kept in KWLinkedList. For Kwlinkedlist the time complexity for add and get method are constant.

```
@Override
public int size() {
	return this.size;
}

private void addLast(E data) {
		if(head == null) {
			head = tail = new Node(data);
		}else {
			Node temp = new Node(data);
			tail.next = temp;
			temp.prev = tail;
			tail = temp;
		}
		size++;
	}
```

Add method calling addLast which is of constant time O(1) for linked list

```
	@Override
	public boolean add(E data) {
		try {
			addLast(data);
		} catch (Exception e) {
			return false;
		}

		return true;
	}

private int getBranchIndexByName(String branchName) {
	for (int i = 0; i < branches.size(); i++) { ------------n
	    if (branches.get(i).getName().equals(branchName))
	      return i;
	}
	return -1;
    }
```

getBranchIndexByName has complexity O(n).

```
public void addBranch(String branchName) {

       if (branches.size() == 0) {
           branches.add(new Branch(branchName));
       } else {
           for (int i = 0; i < branches.size(); i++) {  ----------- n
              if (getBranchIndexByName(branchName) != -1) { -----------O(n)
                  System.out.println("\nBranch already exists !!! \n");
                  return;
              }
           branches.add(new Branch(branchName));
    }
```

So for addBranch method the best time complexity is Ω(1) in the if check. The worst case complexity is n*O(n) or O (n^2).

Remove method in KWlinkedList

```
@Override
       public E remove(int index) {
              if(index < 0 || index >= size())throw new IndexOutOfBoundsException();
              if(index == 0) {
                     E oldData = head.data;
                     removeFirst();
                     return oldData;
              }
              else if(index == size()-1) {
                     E oldData = tail.data;
                     removeLast();
                     return oldData;
              }
              else {
                     Node node = getNodeByIndex(index); -------O(n)
                     E oldData = node.data;
                     Node prev = node.prev;
                     Node next = node.next;
                     node = null;
                     prev.next=next;
                     next.prev=prev;
                     size--;
                     return oldData;
              }
       }
```

The remove method has worst case complexity O(n)

```
public Branch deleteBranch(String branchName) {
       int searchBranchIndex = getBranchIndexByName(branchName); -------------O(n)
       if (searchBranchIndex != -1) {
           return branches.remove(searchBranchIndex);------O(n)
       }
       return null;
    }
```

- The best case complexity is Ω(n) and the worst case complexity is O (n^2) .

```
private int getBranchIndexById(int branchId) {
       for (int i = 0; i < branches.size(); i++) {
           if (branches.get(i).getId() == branchId) {
              return i;
           }
       }
       return -1;
    }
```

- getBranchIndexById has time complexity O (n).

```
private String getBranchNamebyId(int id) {
       for (int i = 0; i < branches.size(); i++) {-----n
           if (branches.get(i).getId() == id) ----O(1)
              return branches.get(i).getName();-------O(1)
       }
       return null;
    }
```

- getBranchNamebyId has complexity O(n)

```
public boolean addBranchEmployee(String name, String surName, int branchId) {

       String branchName = getBranchNamebyId(branchId); ------O(m)

       if (branchName != null) {
           int branchIndex = getBranchIndexById(branchId); -----O(n)
           BranchEmployee employee = new BranchEmployee(name, surName,
UserType.branchEmployee, branches.get(branchIndex)); ----O(1)
           branches.get(branchIndex).getEmployees() -----O(1)
                  .add(employee);
           System.out.println("\nEmployee Id : "+employee.getId()+" (need this id to
login later)\n");
           return true;
       } else {
           return false;
       }

    }
```

- The best case complexity is Ω(n) and worst case is O(m+n).

```
private int getBranchEmployeeIndexById(int branchIndex, int id) {
      for (int i = 0; i < branches.get(branchIndex).getEmployees().size(); i++) {
          if (branches.get(branchIndex).getEmployees().get(i).getId() == id)
            return i;
      }
      return -1;
       }
```

- getBranchEmployeeIndexById O(n)

```
public BranchEmployee deleteBranchEmployee(int branchId, int employeeId) {

          int branchIndex = getBranchIndexById(branchId); -------O(m)
          if(branchIndex!=-1) {
            int employeeIndex = getBranchEmployeeIndexById(branchIndex,
employeeId);--------O(n)
                if (employeeIndex != -1) {

                   return
branches.get(branchIndex).getEmployees().remove(employeeIndex);----O(p)
                } else {
                   System.out.println("\n Invalid Employee Id !!! \n");
                }
          }else {
            System.out.println("\n Invalid branchId !!! \n");
          }



      return null;
    }
```

- The best case time complexity is Ω(m) the worst case complexity is O(m+n+p) as there are three individual blocks.

```
public void seeBranches() {
    System.out.println("\t\t---List of Branches---\n");
    for(int i = 0 ; i < branches.size();i++) {
      System.out.println("Branch Id : " + branches.get(i).getId() + "\t  Branch Name
:  " + branches.get(i).getName());
    }
      System.out.println();
    }
```

- Time complexity is θ(n)

```
public boolean seeBranchEmployees() {
    int count = 0 ;
      for (int i = 0; i < branches.size(); i++) { -----------n
          for (BranchEmployee employee : branches.get(i).getEmployees()) { ------m
             count++;
             System.out.println(employee);
          }

      }
      if(count == 0) {
             System.out.println("\nNo Branch employees!!!. Try adding them
first\n");
             return false;
      }
      else System.out.println();
      return true;
       }
```

- The best and worst case are same . So time complexity is θ(m*n).

```
public void readMsg() {
      for(int i = 0 ; i < Employee.messages.size();i++) {
                  System.out.println(i+1+" : "+ Employee.messages.get(i));
            }
          }

-time complexity θ(n)
```

Branch Employee :

```
/**
     * function to  print list of all products in a particular branch
     */
      @Override
      public boolean listProducts() {
             System.out.println("\t\t\t\t---List of Products In Your Branch---");
             if(this.assignedBranch.getProducts().size()==0) { --------O(1)
                    System.out.println("\nNo Prducts !!!");
                    return false;
             }
             else
             for(int i = 0 ; i < this.assignedBranch.getProducts().size();i++) {--n
             System.out.println(this.assignedBranch.getProducts().get(i)); ----O(1)
             }
             System.out.println();
             return true;
      }
```

- Best case Ω(1) , worst case O(n)

```
public void customerHistory(int customerId) {
		for(int i = 0 ; i < BranchEmployee.sells.size(); i++) {
			if(BranchEmployee.sells.get(i).getCustomer().customerUniqueId()
== customerId) {-----------n
				System.out.println("\n\t\t---Customer History---");
				//if the uniqueId in sellshistory matches with provided
unique id then show info
				System.out.println("customer id : "+customerId +"\t name :
"+ BranchEmployee.sells.get(i).getCustomer().getCustomerName());
				System.out.println();

	if(BranchEmployee.sells.get(i).getCustomerProducts().size()==0) {
					System.out.println("\nCustomer has no product
!!!\n");
					return;
				}
				for(int j = 0 ; j <
BranchEmployee.sells.get(i).getCustomerProducts().size(); j++) { ---------n

	System.out.println(BranchEmployee.sells.get(i).getCustomerProducts().get(j));
				}
				System.out.println();
			}else {
				System.out.println("\nNo Customer Found with this Id
!!!\n");
			}
		}
	}
```

- Time complexity is $\theta(n^2)$

```
/**
	 * function to add product (increase product quantity)
	 * @param productId takes a integer value
	 * @param amountToAdd takes a integer value
	 * @return
	 */
	public boolean addProduct(int productId,int amountToAdd) {
		if(amountToAdd <= 0) return false;
		for(int i = 0 ; i < this.assignedBranch.getProducts().size(); i++) {

	if(this.assignedBranch.getProducts().get(i).getProductId()==productId) {

	this.assignedBranch.getProducts().get(i).increaseAmount(amountToAdd);
				return true;
			}
		}
		return false;
	}
```

- Time complexity O(n)

```
public void informManager(Message msg) {
		messages.add(msg); -------O(1)
	}
```

Messages is of kwarraylist type whose add method has constant time complexity.

- So time complexity of informmanager is O(1)

```
public boolean sellProduct(int productId,int amount,Customer customer) {
		int productIndex = getProductIndexbyId(productId);----O(n)
		if(productIndex!=-1) {

	if(this.assignedBranch.getProducts().get(productIndex).getAmount() < amount) {
				System.out.println("\nNot enough product in stock !!!\n");
				return false;
			}else {
				Product product =
this.assignedBranch.getProducts().get(productIndex);
				//decrease stock amount
				product.decreaseAmount(amount);-----O(1)
				//add to sells history
				//if customer is already present in sells history
				for(int i = 0 ; i < BranchEmployee.sells.size();i++) {

	if(BranchEmployee.sells.get(i).getCustomer().customerUniqueId() ==
customer.customerUniqueId()) { -----------n

	BranchEmployee.sells.get(i).addToSellsHistory(product, amount);
					return true;
				}
				}
				//if customer is not present in the system
				SellsHistory temp = new SellsHistory(customer);
				temp.addToSellsHistory(product, amount);
				BranchEmployee.sells.add(temp);
			}
			return true;
		}
		System.out.println("\nInvalid Product Id !!!\n");
		return false;
	}

-best case is Ω(n) , worst case is O(n) .So time complexity is θ(n)
```

Customer:

Customer uses hybridlist to keep data. get , add , size these methods has constant time complexity.

```
public List<Product> search(String productName){
		ArrayList<Product> temp = new ArrayList<Product>();
		for(int i = 0 ; i < Admin.branches.size(); i++) {---------n
			for(int j = 0 ; j
<Admin.branches.get(i).getProducts().size();j++) {--------n

	if(Admin.branches.get(i).getProducts().get(j).getProductName().equals(productN
ame)) {

	temp.add(Admin.branches.get(i).getProducts().get(j));
				}
			}
		}
		return temp;
	}
```

- Time complexity is $\theta(n^2)$

```
/**
	 * function to list all the products
	 */
	@Override
	public boolean listProducts() {
		System.out.println("\n---List of All Products---");
		int count = 0 ;
		for(int i = 0 ; i < Admin.branches.size(); i++) {
			for(int j = 0 ; j <
Admin.branches.get(i).getProducts().size();j++) {

	System.out.println(Admin.branches.get(i).getProducts().get(j));
				count++;
			}
		}
		if(count == 0) {
			System.out.println("\nNo products !!!\n");
			return false;
		}else return true;
	}
```

- Time complexity is $\theta(n^2)$

```
public void history() {
		for(int i = 0 ; i < BranchEmployee.sells.size(); i++) {-------n
			if(BranchEmployee.sells.get(i).getCustomer().customerUniqueId()
== this.id) {
				System.out.println("\n\t\t---Your Shopping History---");
				//if the uniqueId in sells history matches with provided
unique id then show info
				System.out.println("customer id : "+this.id +"\t name : "+
BranchEmployee.sells.get(i).getCustomer().getCustomerName());
				System.out.println();

	if(BranchEmployee.sells.get(i).getCustomerProducts().size()==0) {
					System.out.println("\nNo product !!!\n");
					return;
				}
				for(int j = 0 ; j <
BranchEmployee.sells.get(i).getCustomerProducts().size(); j++) {---------m

	System.out.println(BranchEmployee.sells.get(i).getCustomerProducts().get(j));
				}
				System.out.println();
			}else {
				System.out.println("\nNo History Found!!!\n");
			}
		}
	}
```

- Best case time complexity is Ω(n) and worst case complexity is O(m*n)

```
public Branch getBranchByProductId(int productId) {
		for(int i = 0 ; i < Admin.branches.size(); i++) {----m
			for(int j = 0 ; j <
Admin.branches.get(i).getProducts().size();j++) {--------n

	if(Admin.branches.get(i).getProducts().get(j).getProductId() == productId &&
Admin.branches.get(i).getProducts().get(j).getAmount()!=0) {
					return Admin.branches.get(i);
				}
			}
		}
		return null;
	}
```

- GetBranchByProductId has time complexity O(m*n)

```
public boolean buyOnline(int productId,int amount) {
		Branch tempBranch = getBranchByProductId(productId);---O(m*n)
		if(tempBranch == null) return false;
		for(int i = 0 ; i < tempBranch.getProducts().size();i++) {-----p
			if(tempBranch.getProducts().get(i).getProductId() == productId) {
// if product id matches
				if(tempBranch.getProducts().get(i).getAmount()>=amount) {
//if amount matches
```

```
	tempBranch.getProducts().get(i).decreaseAmount(amount); // amount decreased
                                for(int j = 0 ; j < BranchEmployee.sells.size();j++)
{----------q

	if(BranchEmployee.sells.get(j).getCustomer().customerUniqueId() == this.id) {

	BranchEmployee.sells.get(j).addToSellsHistory(tempBranch.getProducts().get(i),
amount);
                                        }
                                }
                                return true;
                        }else {
                                System.out.println("\nProduct Out of Stock !!!\n");
                        }
                }
        }
        return false;
	}
```

- Best case Ω(m*n) , worst case O(m*n+p*q)

```
/**
	 * function to update customer's billing address
	 * @param address takes a string value is the address of the customer
	 * @param phoneNum takes a int value is the phone number of the customer
	 */
	public void updateInfo(String address,int phoneNum) {
		this.address = address;
		this.phoneNum = phoneNum;
		System.out.println("\nUpdated !!!\n");
	}
```

- Time complexity O(1)