Use Cases:

Use Case 1(Sign-in Page):

Use Case ID	U01
Name	Sign-in page
Actor	Employee, rider
Description	If the already hired employees and riders have to login to the system then they first have to fill up this page. This is the login page where the employees and riders have to enter their username and password in order to enter into the system. If they enter wrong username or password then the system shows invalid username or invalid password label.
Flow	Basic Flow: 1: Already added employees or riders enters the system. 2: They want to login into their accounts. 3: In order to get signed in they have to enter their usernames and passwords. 4: Then hit the login button. 5: If the username and password match then they are signed in to their accounts. Alternative Flow: 1: Employees or riders enters the invalid name or password. 2: System checks the name and password. 3: If the data entered is invalid, an error message is displayed. Employees and riders can enter a new name and password again. 4: System recognizes the data and give login to the user.

Use Case 2(Create Account):

Use Case ID	U02
Name	Create account
Actor	Employee, rider
Description	If the new employee or rider have to add in the system, then at first they have to create their accounts. They have to sign up to the system. In order to get signed up they have to fill the username, password and confirm password options. Then press the sign up button and an account of the employee or rider has been created.
Flow	Basic Flow: 1: Employees or riders arrives at the company. 3: Press the create account link label on the login page. 4: A sign up page opens. 5: Fill up the username, password and confirm password options. 6: Press the sign up button. 7: An account of a person has been created. Alternative Flow: 1: Employees or riders creates an account. 2: System gives error about account name is not correct. 3: Employees or riders again enters the account details and creates accounts. 4: System creates a new account for the user and the use case ended.

Use Case 3(Add Rider):

Use Case ID	U03
Name	Add Rider
Actor	Employee, Manager
Description	Manager/Employee can add a rider into the system where he has to enter his name, email, phone number, salary and vehicle number. Then press the add button. A new rider would be added to the system.
Flow	**Basic Flow:** 1: A new rider arrives at distribution company. 2: He is hired. 3: Then he is added to the system. 4: The employee enters the rider's name, email, phone number, salary and vehicle number to the system. 5: The employee hits the add button and a new rider is added to the system. **Alternative Flow:** 1: Manager/Employee enters the whole data of rider. 2: Manager/Employee writes something wrong in entering the data. 3: Manager/Employee can remove the data and enter the correct data. 4: Manager/Employee enters wrong datatype variable. There will be exceptions and System do not except the data written.

Use Case 4(Edit Rider):

Use Case ID	U04
Name	Edit Rider
Actor	Employee, Manager
Description	Manager/Employee can edit a rider of the system that was previously added to the system. If the rider's name, email, phone number, salary or vehicle number has to be changed then the employee has to use this edit rider option where he can change the rider's information and press the edit button to save this new information to the system.
Flow	**Basic Flow:** 1: A rider was added to the system previously. 2: we can see all rider's information in a grid view table. 3: If we want to change a rider's information then press the edit button against the information of the respective rider. 4: If the rider's name, email, phone number, salary or vehicle number has to be changed then the employee will use this case. 5: Employee re-enters the information needed to be changed. 6: Employee hits the edit button and the rider's edited information would be added to the system. 7: The newly entered information would then be updated against the respective rider into the system. **Alternative Flow:** 1: Manager/Employee enters wrong datatype variable. There will be exceptions and System do not except the data written. 2: Manager/Employee again enters the correct data and click on the edit button. 3: System will save the edited data. 4: This new edit data is accessed everywhere on System.

Use Case 5(Remove Rider):

Use Case ID	U05
Name	Remove Rider
Actor	Employee, Manager
Description	Manager/Employee can remove a rider from the system that was a part of the system. A grid view table is shown in which all rider's information is shown. In order to remove a rider from the system. Press the remove button and the rider was deleted from the system.
Flow	**Basic Flow:** 1: The Employee wants to delete a rider from the system. 2: He can see all rider's information in a grid view table. 3: If a rider has to be removed from the system then the employee presses the remove button. 4: The respective rider and his information is deleted from the system. **Alternative Flow:** 1: Manager/Employee deletes the rider's data and check the confirmation of deletion. 2: System do not delete the rider's data due to slow down of software or other reasons. 3: Manager/Employee again deletes the rider's data. System deletes it. The use case ends.

Use Case 6(Add Order):

Use Case ID	U06
Name	Add Order
Actor	Employee, Manager
Description	Manager/Employee is shown a table in which all the orders placed were shown. If the employee wants to enter a new order, he has to press the add button and add order page opens where he enters the customer's name, product quantity, Order Id, customer's area and date of order placed.
Flow	**Basic Flow:** 1: A grid view table of all the orders placed is shown to the employee. 2: He wants to add a new order. 3: He presses the add button 4: An Add Order page opens where the employee enters customer's name , product quantity, Order Id, customer's area and date of order placed. 5: Then presses the add button and a new order is placed. **Alternative Flow:** 1: Manager/Employee enters the whole data of order. 2: Manager/Employee writes something wrong in entering the data. 3: Manager/Employee can remove the data and enter the correct data. 4; Manager/Employee enters wrong datatype variable. There will be exceptions and System do not except the data written.

Use Case 7(Edit Order):

Use Case ID	U07
Name	Edit Order
Actor	Employee, Manager
Description	Manager/Employee is shown a table in which all the orders placed were shown. If the employee wants to edit an order placed before, he has to press the edit button, the edit order page opens where he re-enters the customer's name, product quantity, Order Id, customer's area or date of order placed which is to be changed.
Flow	**Basic Flow:** 1: A grid view table of all the orders placed is shown to the employee. 2: He wants to edit an order. 3: He presses the edit button 4: An Edit Order page opens where the employee re-enters customer's name , product quantity, Order Id, customer's area and date of order placed which is to be changed. 5: Then presses the edit button and the changes made to the order placed is then shown. **Alternative Flow:** 1: Manager/Employee enters wrong datatype variable. There will be exceptions and System do not except the data written. 2: Manager/Employee again enters the correct data and click on the edit button. 3: System will save the edited data. 4: This new edit data is accessed everywhere on System

Use Case 8(Remove Order):

Use Case ID	U08
Name	Remove Order
Actor	Employee, Manager
Description	Manager/Employee is shown a table in which all the orders placed were shown. If the employee wants to delete an order, he has to press the remove button. The order placed against that remove button will be deleted from the order list.
Flow	**Basic Flow:** 1: A grid view table of all the orders placed is shown to the employee. 2: He wants to delete an order. 3: He presses the remove button 4: The order against this remove button will be deleted from the order list. **Alternative Flow:** 1: Manager/Employee deletes the order's data and check the confirmation of deletion. 2: System do not delete the order's data due to slow down of software or other reasons. 3: Manager/Employee again deletes the order's data. System deletes it. The use case ends.

Use Case 9(Add Employee):

Use Case ID	U09
Name	Add Employee
Actor	Manager
Description	Manager can add employee into the system where he has to enter his name, phone number, salary and Id. Then press the add button. A new rider would be added to the system.
Flow	**Basic Flow:** 1: A new employee arrives at distribution company. 2: He is hired. 3: Then he is added to the system. 4: The manager enters the employee's name, phone number, salary and Id to the system. 5: The manager hits the add button and a new employee is added to the system. **Alternative Flow:** 1: Manager enters the whole data of employee. 2: Manager writes something wrong in entering the data. 3: Manager can remove the data and enter the correct data. 4: Manager enters wrong datatype variable. There will be exceptions and System do not except the data written.

Use Case 10(Edit Employee):

Use Case ID	U10
Name	Edit Employee
Actor	Manager
Description	Manager can edit employee of the system that was previously added to the system. If the employee's name, phone number, salary or Id has to be changed then the manager has to use this edit employee option where he can change the employee's information and press the edit button to save this new information to the system.
Flow	**Basic Flow:** 1: A employee was added to the system previously. 2: we can see all employee's information in a grid view table. 3: If we want to change employee's information then press the edit button against the information of the respective employee. 4: If the employee's name, phone number, salary or Id has to be changed then the manager will use this case. 5: Manager re-enters the information needed to be changed. 6: Manage hits the edit button and the employee's edited information would be added to the system. 7: The newly entered information would then be updated against the respective employee into the system. **Alternative Flow:** 1: Manager enters wrong datatype variable. There will be exceptions and System do not except the data written. 2: Manager again enters the correct data and click on the edit button. 3: System will save the edited data. 4: This new edit data is accessed everywhere on System.

Use Case 11(Remove Employee):

Use Case ID	U11
Name	Remove Employee
Actor	Manager
Description	Manager can edit employee of the system that was previously added to the system. If the employee's name, phone number, salary or Id has to be changed then the manager has to use this edit employee option where he can change the employee's information and press the edit button to save this new information to the system.
Flow	**Basic Flow:** 1: The Manager wants to delete employee from the system. 2: He can see all employee's information in a grid view table. 3: If employee has to be removed from the system then the manager presses the remove button. 4: The respective employee and his information is deleted from the system. **Alternative Flow:** 1: Manager deletes the employee's data and check the confirmation of deletion. 2: System do not delete the employee's data due to slow down of software or other reasons. 3: Manager again deletes the employee's data. System deletes it. The use case ends.

Use Case 12(Add Product):

Use Case ID	U12
Name	Add Product
Actor	Manager
Description	Manager can add products into the system where he has to enter name, stock, threshold, price, total product sold, expiry date and size of product. Then press the add button. A new product would be added to the system.
Flow	**Basic Flow:** 1: A new product created at distribution company. 2: Then it is added to the system. 3: The manager enters the name, stock, threshold, price, total product sold, expiry date and size of product to the system. 4: The manager hits the add button and a new product is added to the system. **Alternative Flow:** 1: Manager enters the whole data of product. 2: Manager writes something wrong in entering the data. 3: Manager can remove the data and enter the correct data. 4: Manager enters wrong datatype variable. There will be exceptions and System do not except the data written.

Use Case 13(Edit Product):

Use Case ID	U13
Name	Edit Product
Actor	Manager
Description	Manager can edit a product of the system that was previously added to the system. If the name, stock, threshold, price, total product sold, expiry date and size of product has to be changed then the manager has to use this edit product option where he can change the product's information and press the edit button to save this new information to the system.
Flow	**Basic Flow:** 1: A product was added to the system previously. 2: We can see all product's information in a grid view table. 3: If we want to change a product's information then press the edit button against the information of the respective product. 4: If the name, stock, threshold, price, total product sold, expiry date and size of product has to be changed then the manager will use this case. 5: Manager re-enters the information needed to be changed. 6: Manager hits the edit button and the product's edited information would be added to the system. 7: The newly entered information would then be updated against the respective product into the system. **Alternative Flow:** 1: Manager enters wrong datatype variable. There will be exceptions and System do not except the data written. 2: Manager again enters the correct data and click on the edit button. 3: System will save the edited data. 4: This new edit data is accessed everywhere on System.

Use Case 14(Remove Product):

Use Case ID	U14
Name	Remove Product
Actor	Manager
Description	Manager can remove a product from the system that was a part of the system. A grid view table is shown in which all product's information is shown. In order to remove a product from the system. Press the remove button and the product was deleted from the system.
Flow	**Basic Flow:** 1: The Manager wants to delete a product from the system. 2: He can see all product's information in a grid view table. 3: If a product has to be removed from the system then the manager presses the remove button. 4: The respective product and his information is deleted from the system. **Alternative Flow:** 1: Manager deletes the product's data and check the confirmation of deletion. 2: System do not delete the product's data due to slow down of software or other reasons. 3: Manager again deletes the product's data. System deletes it. The use case ends.

Use Case 15(Manage Expenses):

Use Case ID	U15
Name	Manage Expenses
Actor	Manager
Description	Manager can record all expenses, profits, loses and bonus occurs at the company.
Flow	**Basic Flow:** 1: The manager records the expenses of riders such as vehicle given to them, fuels used by vehicle. 2: Manager also records the salary given to riders and employees. 3: Manager record all bonus to riders or employees and all profits, loses occurs at company.

Use Case 16(short path of map):

Use Case ID	U16
Name	short path of map
Actor	Employee
Description	Employee gives shortest path given on the map to the riders. So that rider can deliver the orders to customers by using shortest path and there will be less usage of fuels by using shortest path.
Flow	**Basic Flow:** 1: Employee opens the Map UI. 2: Employee enters the location by giving values to longitude and latitude of map. 3: It gives the shortest path to riders. 4: Riders uses that path and delivers the orders to the customers. **Alternative Flow:** 1: Employee gives wrong path to riders and then correct it by redirecting again the path. 2: System gives the correct path that shows to riders and he delivers the orders.

Use Case 17(Send Email):

Use Case ID	U17
Name	Send Email
Actor	Manager, Employee
Description	Manager/Employee can send email to the customer for thanking them to purpose stock from their company. They can also send email to customers: if customers any previous record of payment is left, if the payment is clear or if there is any problem in sending late orders to customer.
Flow	**Basic Flow:** 1: Manager/Employee opens the Email Sending UI. 2: Manager/Employee enters the name to send email to that customer. 3: Manager/Employee writes Subject ad body of email for why they are sending email to customer. 4: Manager/Employee enters the send button, a message is shown that the email is send and it delivered to the respective customer. **Alternative Flow:** 1: Manager/Employee open UI to send email to customer. 2: Manager/Employee enters wrong email of customer. 3: System shows the error that email is incorrect as there is no customer email present that manager or employee writes. 4: Manager/Employee see the correct email of that customer and send email. 5: System verifies the email. 6: A message is shown that the email is send.

Use Case 18(make graph):

Use Case ID	U18
Name	make graph
Actor	Manager
Description	Manager can see the financial report of profits or loss in Graph form where the percentages is given for profits and loss in company expenses management.
Flow	**Basic Flow:** 1: Manager opens financial report and wants to see analytical report of finances. 2: Manager clicks on the analytical button and a graph is shown where visualized profits or losses are put together.

Use Case 19(Deliver Ordered Products):

Use Case ID	U19
Name	Deliver Ordered Products
Actor	Rider
Description	Riders have completely visible data of orders. Riders have total number of ordered products that delivers to the customers. If ordered products are delivered, a check box is ticked by the rider.
Flow	**Basic Flow:** 1: Rider's work is to deliver all the ordered products of customers. 2: System presents all ordered products done by the customers. 3: Rider checkout all orders. 4: Rider starts from the top most orders of customer and delivers to that customer. 5: Customer receives the orders. 6: Rider tick on the check box to assure that the product is delivered. 7: System shows that the ordered product is delivered to the customer. **Alternative Flow:** 1: If the order is not delivered, product again added to the orders list of all customers.

Use Case 20(Show location of Customer to deliver products):

Use Case ID	U20
Name	Show location of Customer to deliver products
Actor	Rider
Description	Rider watch the location of customer by map. Employee gives location to the rider and rider delivers the ordered product to customer. Employee have complete location of rider.
Flow	**Basic Flow:** 1: Rider starts a new sale. 2: Employer gives location of customer. 3: System shows the location of customer to the rider. 4: Rider watch the location by map and delivers the ordered products to customer. **Alternative Flow:** 1: If the location is not correct, rider get correct location from Employee. 2: System updated the location.

Use Case 21(update payments of Customer):

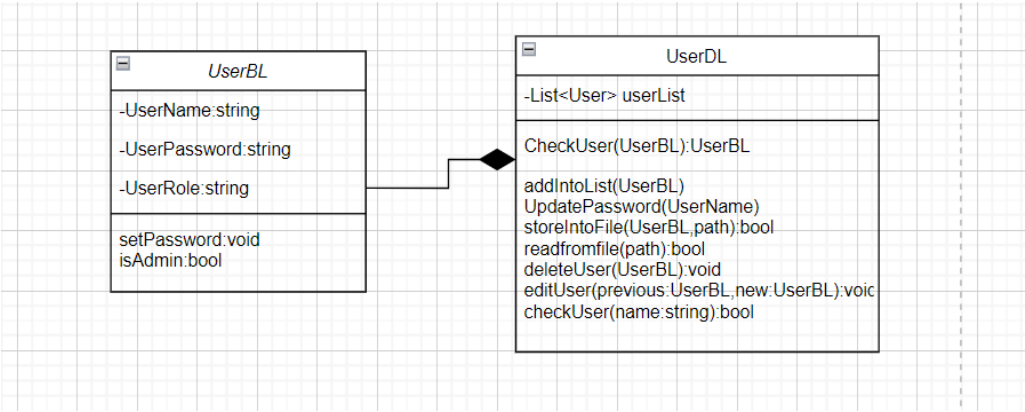
Use Case ID	U21
Name	update payments of Customer
Actor	Rider
Description	Rider updates the payment by just tick on check box, only if it is cash payment.
Flow	**Basic Flow:** 1: Rider delivers the order. 2: If the payment is cash, the rider clicks on check box. So that the customer's payment is checked as received. **Alternative Flow:** 1: If payment is not on cash, rider will do nothing.

Classes:

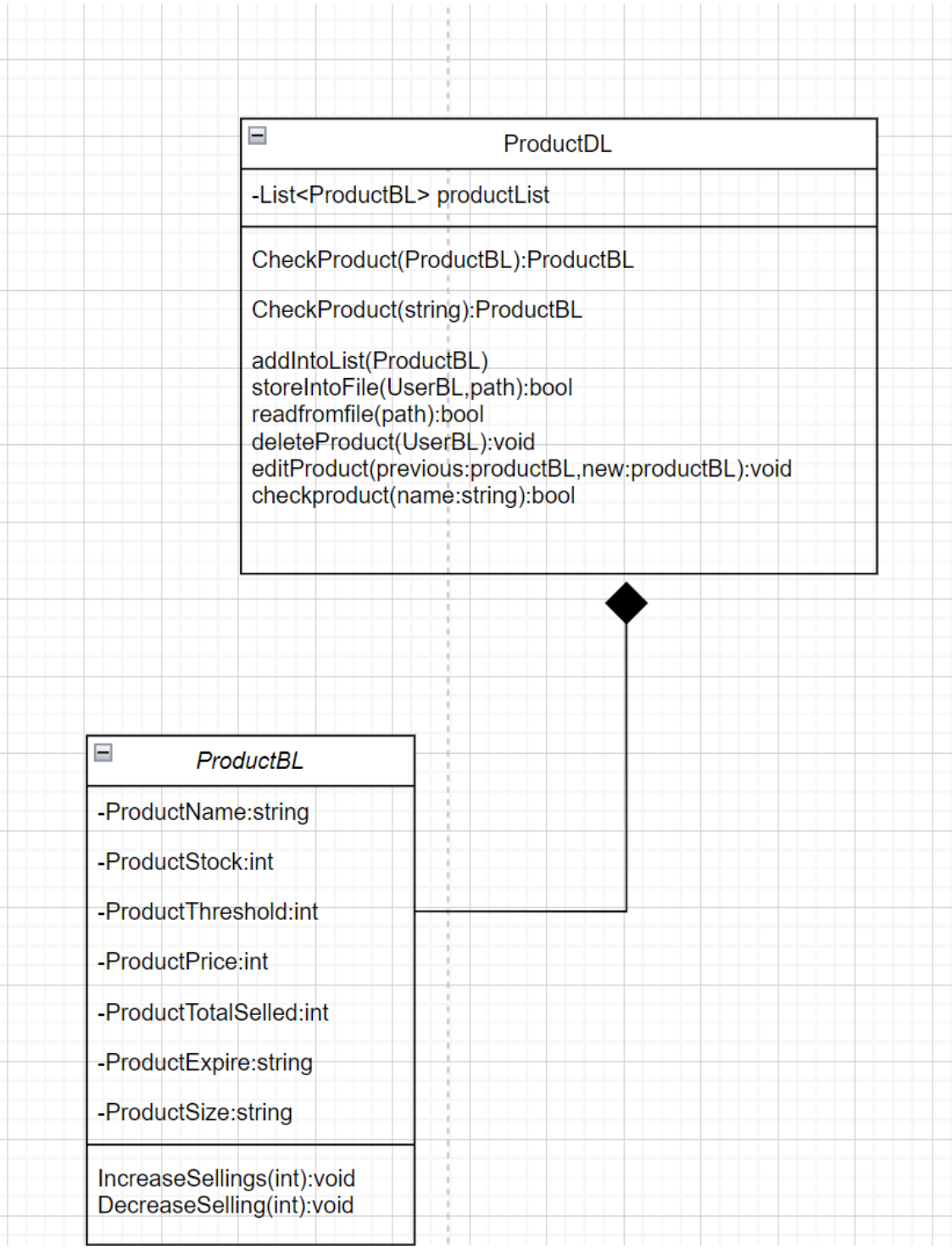
Classes Name	Software /Domain	Is Abstract (Yes/No)	Is Singleton (Yes/No)	Is the class will has parametrized constructor(Yes/No)
UserBL	Domain	No	No	Yes
UserDL	Software	No	Yes	No
CustomerBL	Domain	No	No	Yes
CustomerDL	Software	No	Yes	No
EmployeeBL	Domain	No	No	Yes
EmployeeDL	Software	No	Yes	No
OrderBL	Domain	No	No	Yes
OrderDL	Software	No	Yes	No
ProductDL	Domain	No	Yes	No
ProductBL	Domain	No	No	Yes
RiderBL	Domain	No	No	Yes
RiderDL	Domain	No	Yes	No
area	Software	No	No	Yes
dijkstraAlogrithm	Software	No	No	No
ExpenseMangement	Domain	No	No	No

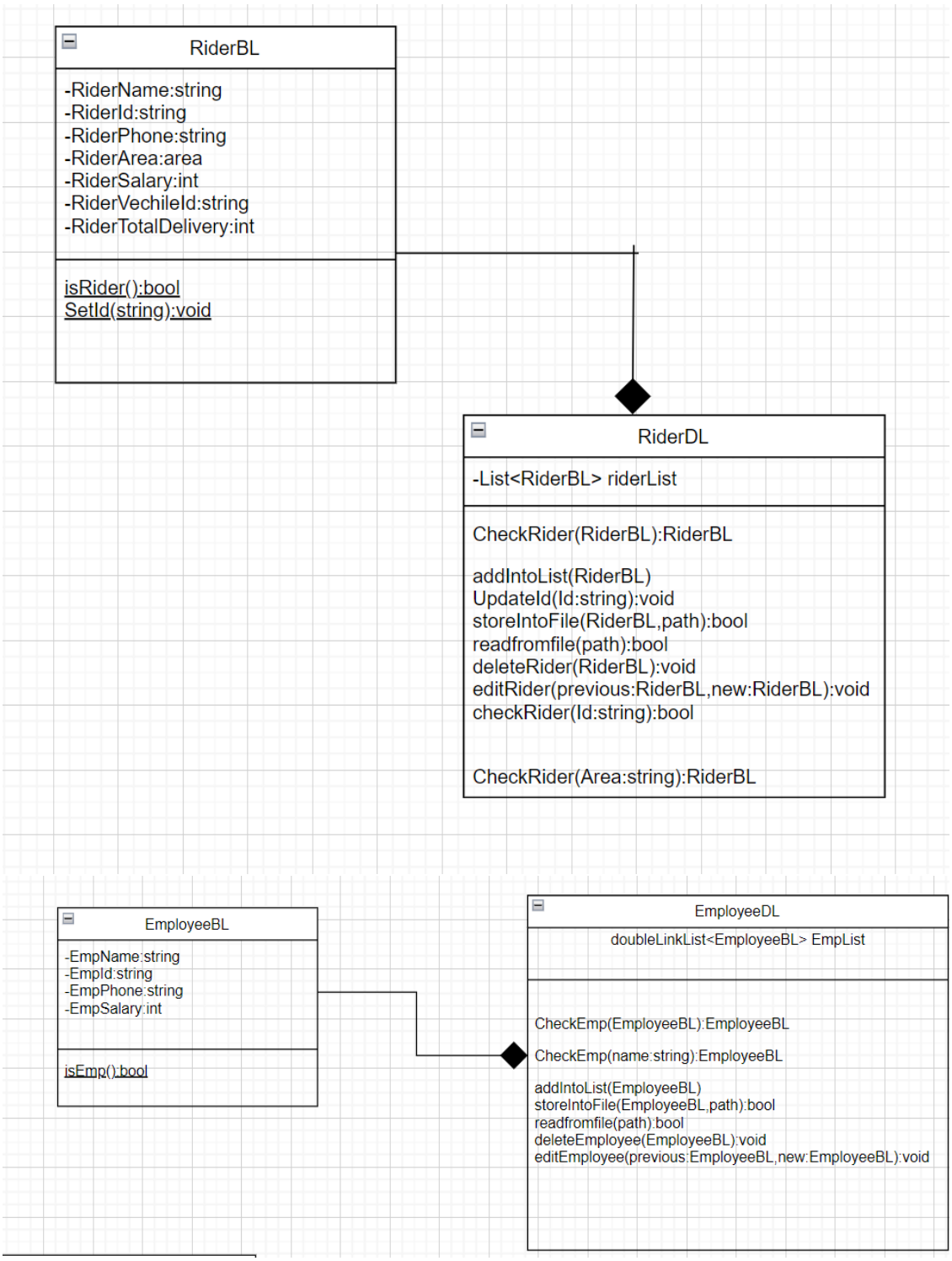
Object Oriented Features:

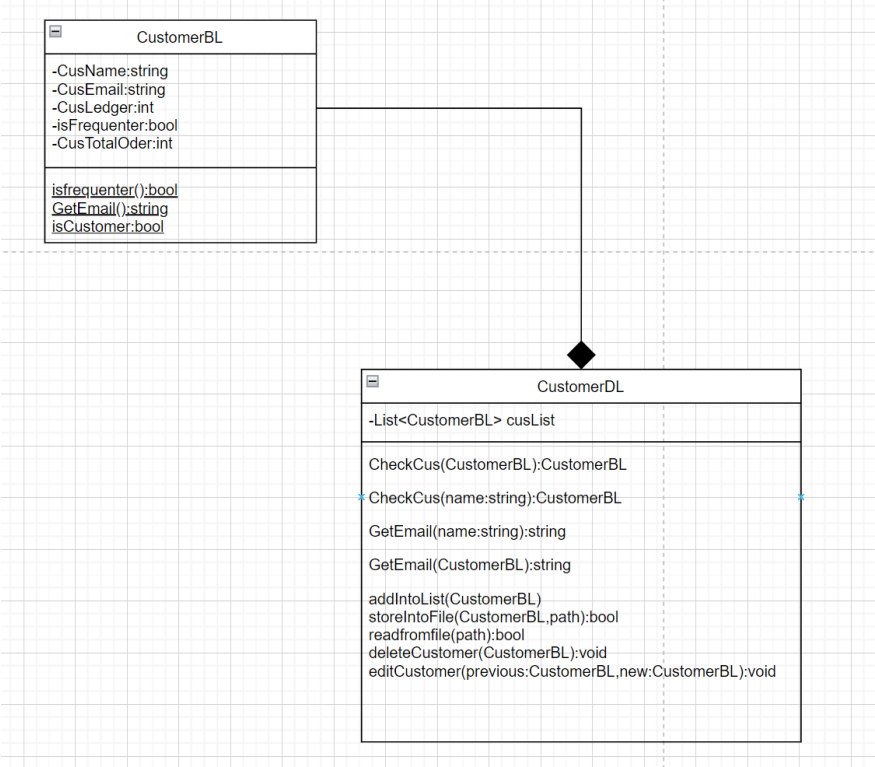
Composition:



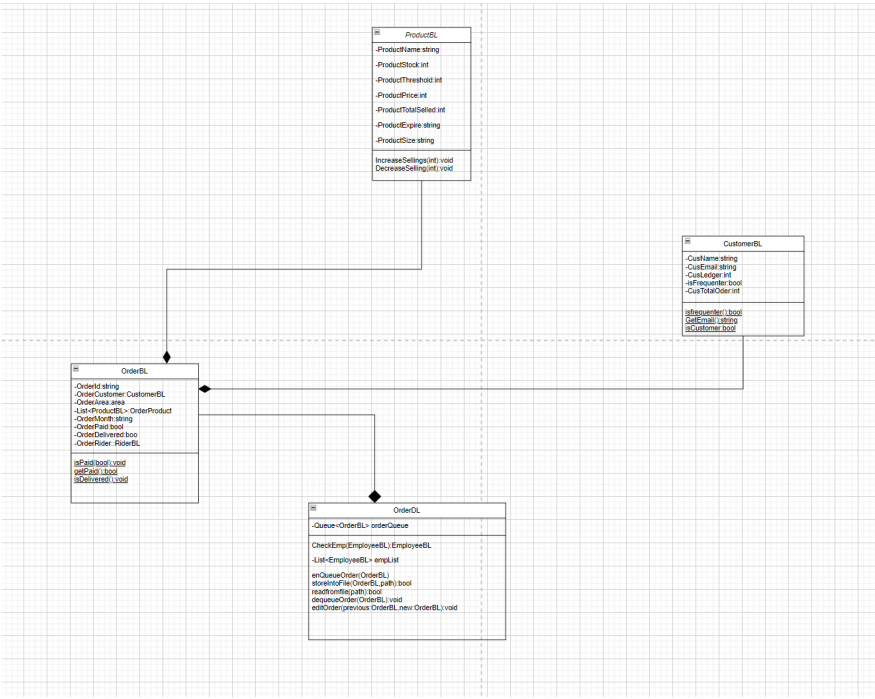
LikeWise UserBL and UserDL there are several examples are there such as ProductDL , RiderDL, EmployeeDL and CustomerDL acting similar to UserDL handling data logics for ProductBL, RiderBL EmployeeBL and CustomerBL respectively.


ProductDL
-List<ProductBL> productList
CheckProduct(ProductBL):ProductBL
CheckProduct(string):ProductBL
addIntoList(ProductBL)
storeIntoFile(UserBL,path):bool
readfromfile(path):bool
deleteProduct(UserBL):void
editProduct(previous:productBL,new:productBL):void
checkproduct(name:string):bool
ProductBL
-ProductName:string
-ProductStock:int
-ProductThreshold:int
-ProductPrice:int
-ProductTotalSelled:int
-ProductExpire:string
-ProductSize:string
IncreaseSellings(int):void
DecreaseSelling(int):void


RiderBL
-RiderName:string
-RiderId:string
-RiderPhone:string
-RiderArea:area
-RiderSalary:int
-RiderVechileId:string
-RiderTotalDelivery:int
isRider():bool
SetId(string):void
RiderDL
-List<RiderBL> riderList
CheckRider(RiderBL):RiderBL
addIntoList(RiderBL)
UpdateId(Id:string):void
storeIntoFile(RiderBL,path):bool
readfromfile(path):bool
deleteRider(RiderBL):void
editRider(previous:RiderBL,new:RiderBL):void
checkRider(Id:string):bool
CheckRider(Area:string):RiderBL
EmployeeBL
-EmpName:string
-EmpId:string
-EmpPhone:string
-EmpSalary:int
isEmp():bool
EmployeeDL
doubleLinkList<EmployeeBL> EmpList
CheckEmp(EmployeeBL):EmployeeBL
CheckEmp(name:string):EmployeeBL
addIntoList(EmployeeBL)
storeIntoFile(EmployeeBL,path):bool
readfromfile(path):bool
deleteEmployee(EmployeeBL):void
editEmployee(previous:EmployeeBL,new:EmployeeBL):void

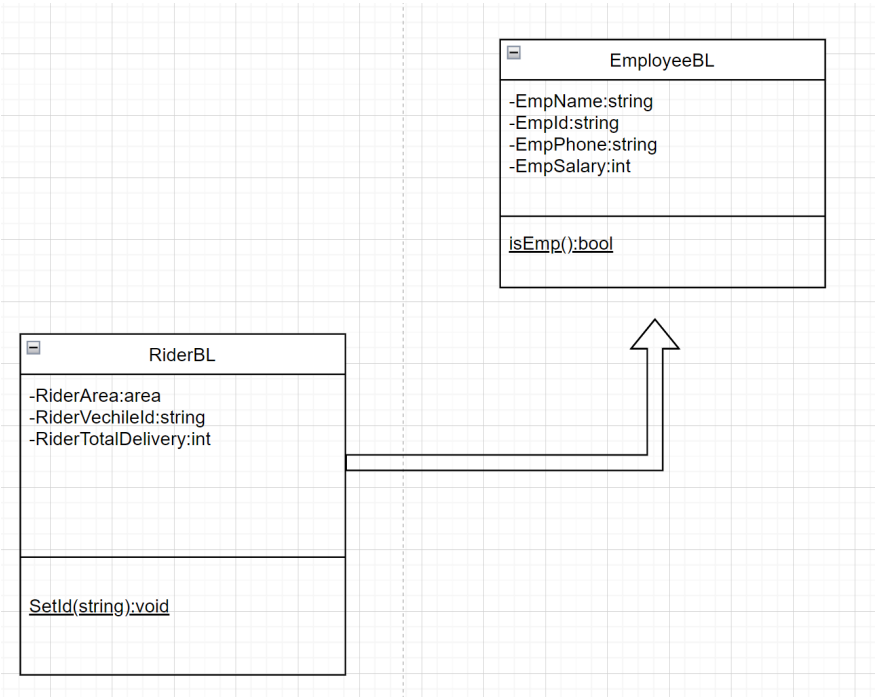


The OrderBL is composited by ProductBL, the class containing data about products, CutsomerBL containing data



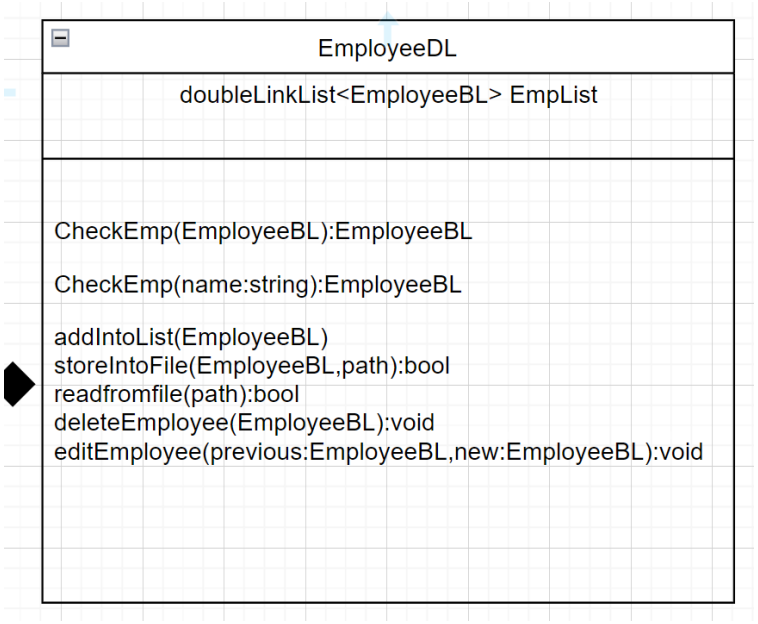
Inheritance:

The rider is also an employee but it has some extra features so the RiderBL, class having information about Rider inherits the EmployeeBL.



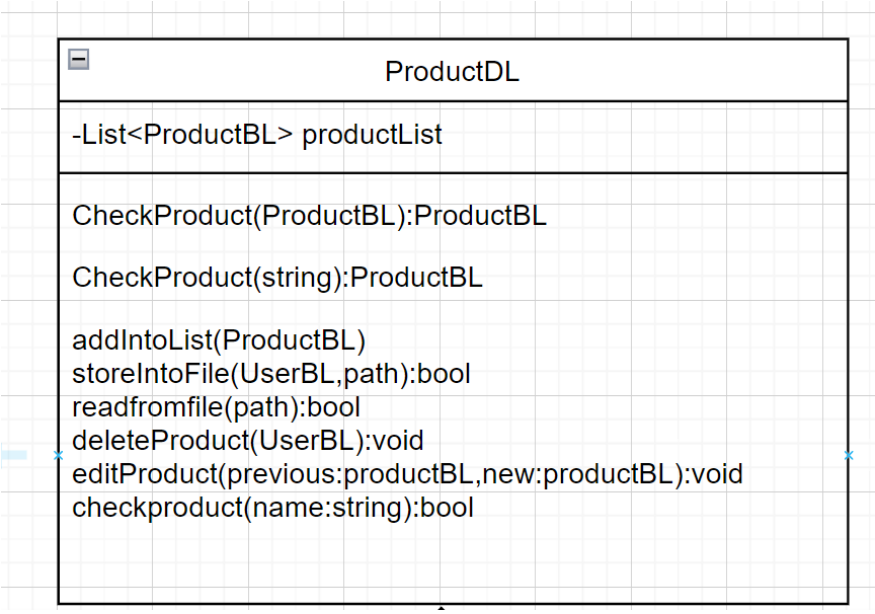
Polymorphism:

The class having function CheckEmpty which can check the Employee Existence function by the object or through its name both the function have same return type.

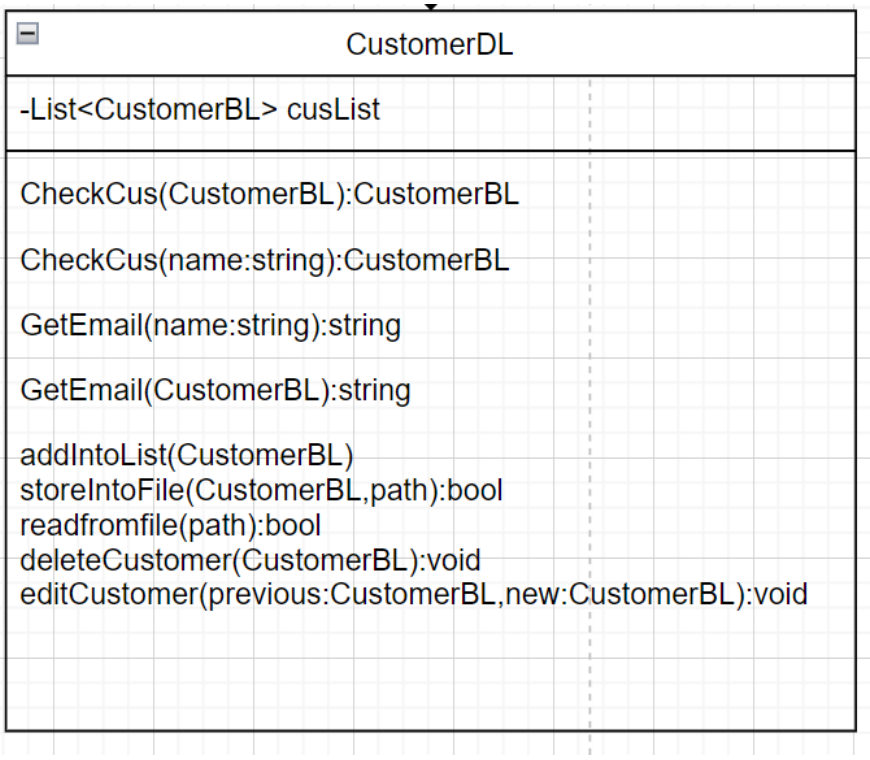


Similarly the same case is in ProductDL searching product by its name or object through function

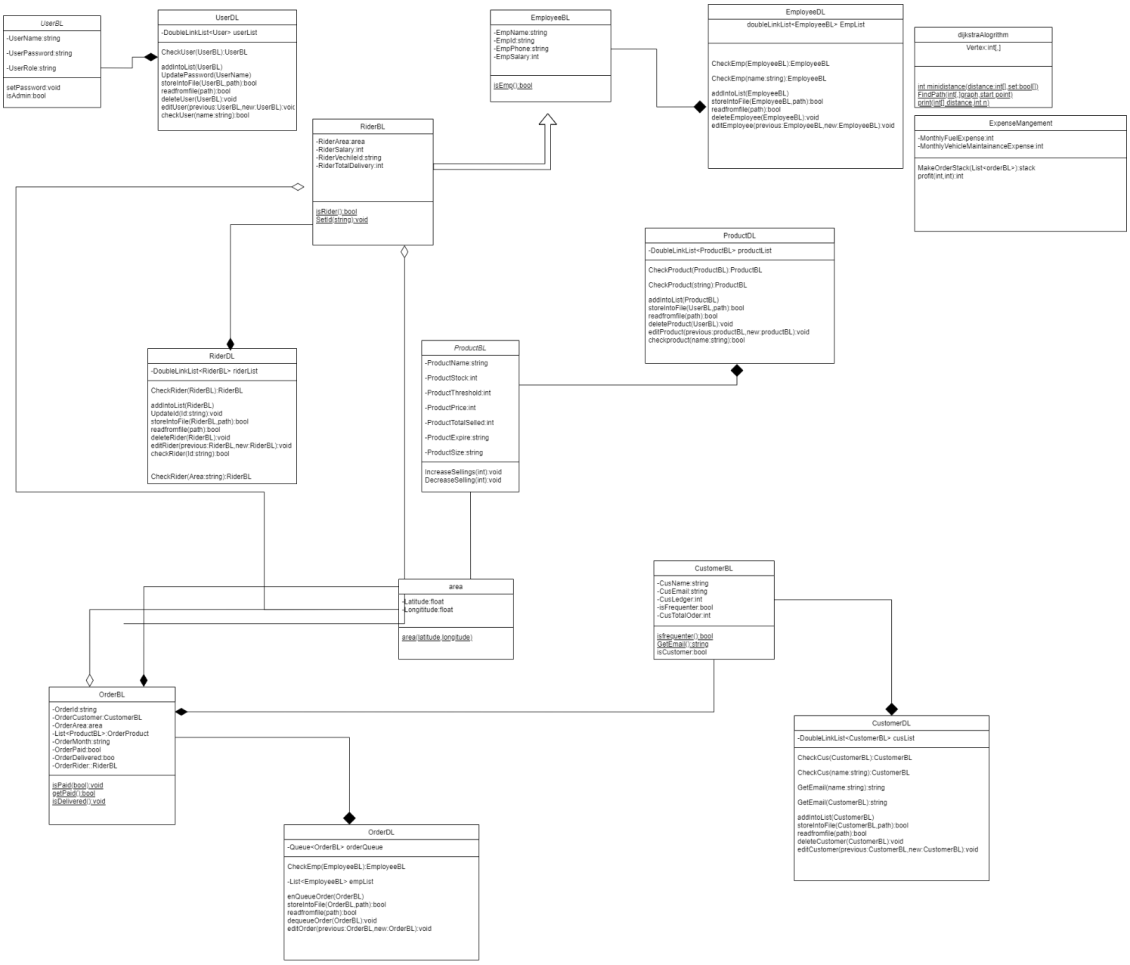
CheckProduct.



The CustomerDL have the similar searching option but also GetEmail which can be used in the forms to access the email, it can be access by the name of customer which is string, also through CustomerBL object.



Detailed Object Oriented Design:



Data Structure:

Use Case Id	Data Structures Used	Justification for the usage of data structure
U01,U04,U05,U13	AVL Tree	The Doubly LinkedList has stored the data, which is converted into AVL tree and searching is applied as its time complexity is best as compared to others, which is O(lg n).
U02,U03,U05,U09,U11 U12,U14	Doubly Link List	The data of newly added data are stored in the doubly LinkedList as insertion and deletion takes O(1) , for searching it is converted into AVL tree.
U06,U07,U08	Queue	The data of newly added riders and already added are stored in the Queue as insertion in O(1)and deletion, searching takes O(n), Also it will follow FIFO policy.

Exceptions:

Type of Exception	Why this exception will occur	Use Case Id in which exception could be occurred	How you will handle the exception
Invalid User	The Username/Password enter doesn't match the data or the password doesn't match.	U01	Displaying which of both username or password is wrong. If the user the user doesn't remember the password he/she can go to forgot password form.
Repetition	The Id already assigned is being using again.	U03,U04,U06,U07, U09	The data will checked if it already exists, the user can be given a chance to change the Id.There should unique ID in every case.
Unavailability	The order is placed for the product which is not available or maybe there is no rider.	U06,U07	The employee has the option to refill the stock and hire all riders.
Host Error	The email sent other than the email.	U17	The email should be only email here will be restriction.

Data Storage:

The Data are stored in .csv file, our employee, order, customer, user, product that we be stored in different files. Employee.csv will store name, id, phone number, salary of the employee. Rider.csv will store name, id, phone number, salary of the employee, area and vehicle id assigned to him and total deliveries. Order.csv will store Order id, info of the customer, area, product and rider who will deliver the order using object oriented mapping month of order, bool if it is paid, status of delivery. User.csv will store name, password and role of the user. Customer.csv will store name, email, ledger, if he/she is frequenter in a bool and total orders. The data will be loaded from the file using stream reader, stored in the respected lists. If any changes are made the list is rewritten.

Email Sending:

The Email will be send to customers for "total amounts of order".
Sample:
Subject: "Your Order has been placed

Hello Customer,
Thank you for your order. We appreciate your business and will be thrilled to send you [Products Ordered] as soon as possible. An email with tracking information will be sent to you once your order has shipped. Your total bill is 70,000.
Thanks again, and we look forward to seeing you soon.
UMM Distribution Company

Project Plan:

Use Case Id	Use Case Name	Member Name	Estimated Completion Date
U01,U23	Sign In page	Mahnoor Hassan	28-11-2022
U02	Create account	Mahnoor Hassan	28-11-2022
U03	Add Rider	Uswa Arif	28-11-2022
U04	Edit Rider	Uswa Arif	28-11-2022
U05	Remove Rider	Uswa Arif	28-11-2022
U06	Add Order	Mutaiba Mohsin	28-11-2022
U07	Edit Order	Mutaiba Mohsin	28-11-2022
U08	Remove Order	Mutaiba Mohsin	28-11-2022
U09	Add Employee	Mahnoor Hassan	29-11-2022
U10	Edit Employee	Mahnoor Hassan	29-11-2022
U11	Remove Employee	Mahnoor Hassan	29-11-2022
U12	Add Product	Mutaiba Moshin	29-11-2022
U13	Edit Product	Mutaiba Mohsin	29-11-2022
U14	Remove Product	Mutaiba Mohsin	29-11-2022
U15	Manage Expense	Uswa Arif	2-12-2022
U16,20	Map Implementation	Mutaiba Mohsin	3-12-2022
U17	Gmail	Uswa Arif	3-12-2022
U18	Graph Implementation	Mutaiba Mohsin	4-12-2022
U19,U23	Deliver Ordered Products	Uswa Arif	7-12-2022
U21	Update Payment	Mahnoor Hassan	7-12-2022
U22	Searching Implementation	Mahnoor Hassan, Mutaiba Mohsin, Uswa Arif	8-12-2022
U24	Report	Mahnoor Hassan, Uswa Arif	9-12-2022

Analytical Reports:

We will show the employee the success of sell of products which product sold highest, also the profit earned within each month, the riders progress showing number of deliveries done by the rider. This all data will be maintained side by side while any change is made, the data will be shown in the graphs Line chart, bar chart and pie chart the user can choose which type of graph he wants.
The data can be sorted providing number of sales highest from which frequenter, allows the company to make strong relations with that retailer and where there are less sales to identify the reason and improve it. The data works in best interest of the workers.