University Of Engineering and Technology,
Lahore
Faculty of Electrical Engineering
Department of Computer Science

CS-261L Term Project

Data Structures and Algorithms

Distribution Management System

Group 01

Shahzaib Irfan *2021-CS-7*
Afraz Butt *2021-CS-12*
Muhammad Hamza *2021-CS-41*

Supervisor: Mr. Samyan Qayyum Wahla

Submission date: 26/11/2022

Table of Contents

List of Figures

List of Tables

Chapter 1

Project Background

1.1 Project Statement

Distribution Management, also known as Supply Chain Management System, are essential for the working of any business, let it be small or large. The difference principally comes in the scale, but micro details remain same. Order management is normally tedious, when done *physically*. What if there was a software / project that could automate all that?
This project is intended to assist all persons involved in a supply chain operation (Shopkeepers, Administrators and Delivery Riders / Truckers) in their operation.

1.2 Description

Managing the flow of goods from a supplier or manufacturer to a point of sale is referred to as distribution management. It is a general phrase that covers a wide range of operations and procedures, including supply chain management, inventory management, warehousing, and packaging.
For distributors and wholesalers, distribution management is an essential component of the business cycle. The ability of an organization to quickly turn over its products affects its profit margins. The future of the company will be brighter as they sell more and make more money. Businesses need a good distribution management system to maintain customer satisfaction and competitiveness.
Distribution Management System (DMS) is also important in recruiting and satisfying clients. It ensures the organization's long-term viability and competitive advantage. It aids in the management of profitable enterprises. Amazon is an excellent example of how to use DMS successfully and efficiently.
The most recent distribution management systems collect and disseminate pertinent information to assess industry potential for growth and competitiveness. Distribution is divided into two categories:

1. Sales Distribution

2. Logistics

Distribution Management has a direct impact on the organization's earnings. To comprehend the significance of DMS, consider the obstacles that sales channels encounter. The organization develops numerous sales and marketing tactics to reach a larger audience. The channels are a way for these things to be sold, but many distributors are tiny and inefficient. They lack both finance and technology. To enter rural areas, more levels of the distribution network must be added, incurring additional expenditures. There is no real-time data on orders, inventories, or claims, and returns result in understocking or overstocking.
This system will also assist businesses to maximise labour and space utilisation, as well as equipment investments, by coordinating and optimising resource usage and material movements. It is intended to support the needs of a worldwide supply chain, including distribution, manufacturing, asset-intensive, and service businesses. Connected consumers want to buy everywhere, fulfil anywhere, and return anywhere in today's dynamic, omni channel fulfilment market. To meet this demand, customers must be able to respond fast. This is done using a distribution system.
This system will also get rid of the old pen-and-paper managing technique, which is time taking and super error prone. It will automate and greatly simplify the whole process and streamline fast working to satisfy client.
This system takes into account three characters of its working and revolves around them. Note that these actors are an abstract level representation of real-world entities. So, one entity might refer to several of its counterparts. For example, an admin might refer to either the owner of the distribution management system or the workers physically responsible for dispatching the goods to the customer. Both come under the heading of admin staff, but their functions and operational scopes are vastly different.

1.2.1 Key Takeaways

1. Distribution management oversees a company's whole supply chain.

2. Distribution management maintains order and client satisfaction. *(In this context, client satisfaction means order fulfillment on time and quick actions on any complaints, if any.)*

1.3 Advantages of the system

1. It keeps things organised. Without a proper management system in place, retailers would be forced to keep stock in their own locations, which is a bad idea, especially if the seller lacks adequate storage space.

2. A distribution management system simplifies things for the consumer as well. It enables them to launch a single application for a wide range of products. Consumers would have to visit multiple locations to get what they need if the system did not exist.

1.4 Project Features

Following are the features offered by this project to its end user.

- Administrator is able to implement inventory control
- Riders and Administrators are able to synchronize and perfectly able to execute and deliver Orders
- Riders are able to take online payments
- Rider is able to navigate to his destination
- All actors are able to analyze their actions

1.5 Technology Stack

IDE	• VS Code • VS • Spyder
Language	• Python • c# • c++

Table 1.1: Table denoting the technology stack we have used

1.6 Project Actors

1.6.1 Primary Actors

- Admin will be the first actor. They will directly supervise the warehouse , the distribution network and its entities. They will be responsible for the overall upkeep in all aspects. This contains managers, production line workers etc.

- Riders are the second actor. They will be responsible for delivering the finished products from warehouse to end user. They are also responsible for taking orders from shopkeepers and synchronizing payments.

1.6.2 Stake Holders

- Government agencies, which have an interest in collecting tax information.

1.7 Use Cases

U01,U19,U20,U21 Concerning Rider use data structure of link list.
U02,U24,U25,U29 concerning shop use data structure of graphs.
U03,U04,U05,U06,U13,U18 concerning orders use data strucuture queue.
U07,U08,U10,U26 concerning products use data structure link list.
U12,U15 concerning Routes involve graph.
U17 concerning Rider Fuel Logs involves usew of Binary Search Tree.

See Justifications of Data Structures for further explanation.

1.7.1 Use Case 01 - Add Rider

Use case 01 - Add rider

Use Case ID	U01
Name	Add Rider
Actor	Admin
Description	The system's administrator can register new riders. The Admin will enter all of the riders' details, including their names, contact information, the regions they will be delivering to, and the orders that have been given to them. The rider will be added to the system and be able to log in once all the necessary information has been entered.
Flow	**Main Success Scenario *(Base Flow)*:** 1. Admin logs in to the system and presses the Add Rider prompt in the menu. 2. There are certain informations that the admin has to add about the rider (*trainee*) in the system. These are: • CNIC • Name • Phone • Address • username • password 3. After the rider is added in the system, his details are stored. **Extensions (*Alternate Flow*):**If at any time, the system fails, then: Admin requests the recovery of system to previous stable state, which includes, *but is not limited t*o, • filling of all information which was filled before failure(*if there was any*) If Add option was toggled, then the respective data must be saved into • the system as usual. This goes without saying that the admin need not re-enter the information again. 1. If the manager leaves some information empty and presses the add button: • Control will not proceed ahead. A informative icon / popup is displayed, showing to the admin where they missed the information. Then upon entering the correct state, the control proceeds ahead. 1-2. CNIC is less than 13 digits or contains a dash, or contains a letter: • Appropriate use of validators will be applied in this case. The control will not proceed ahead if the CNIC is not according to format. 1-3. Phone number contains digits other than 11 in length, or any digit. • Same extension as in case 1-2. , except that in this case, validation digit • length would be 11 instead of 13.

1.7.2 Use Case 02 - Add Shop

Use case 02 - Add Shop

Use Case ID	U02
Name	Add Shop
Actor	Rider
Description	During the journey, a rider encounters shops. Some of these may be their *clients*, but others are not. These shops must be added into the system so that they can be catered to. This task befalls on the rider. It is possible that a single shopkeeper has multiple shops. In this case, only shop data needs to be added.

• Flow	**Main Success Scenario *(Base Flow)*:**1. Rider presses the Add Shop prompt in the menu. 2. Shop details such as: • Name • Area • City • Contact • Location (*From google maps*) and Shopkeeper details such as: • Name • Email are added to the system after being filled in. 3. The location in meridian system (*latitudes and longitudes*) will be filled based on google maps(*backend logic*). 4. Shop is added to the system. **Extensions (*Alternate Flow*):**If at any time, the system fails, then:Admin requests the recovery of system to previous stable state, which includes, *but is not limited t*o, Filling of all information which was filled before failure(*if there was any*) If Add option was toggled, then the respective data must be saved into the system usual. This goes without saying that the admin need not re-enter the information again. 1-1: If Shop data is not entered correctly: 1-1-a) If any data is missing, an informative icon / prompt will be displayed, so that the ridergets aware of the incorrect entr(ies). Appropriate measures such as scrolling to the top most vacantcomponent would be deployed to assist the user in such cases. 1-1-b) If data entered is not valid, such as CNIC(*length 13, no dashes/alphabets*), or phone number(*length 11, no dashes/alphabets , starts with 03*): Appropriate use of validators will be applied in this case. Error messages just below the input prompt will be displayed. 1-2: If shopkeeper data is entered invalid, then error messages will be displayed. 1-3: During location selection from Google Maps, if the external link fails to respond, Add Shop operationwill be halted. 1-3-a): If Google Maps fails to fetch location, the page will be reloaded. The data entered previouslywill be reloaded, and location add prompt will be retoggled. 1-3-b): If Google Maps does not fetch location at all (*out of range*), the prompt will be closed.

1.7.3 Use Case 03 - Checking Order Status

Table 1.5: Checking Order Status, Use Case 3

Use Case ID	U03
Name	Checking Order Status
Actor	Admin
Description	Admin, after being placed the order, must be able to update the order in real world, its status and any related developments. This can either be done using a plain text messages (as in post applications) or can be used as a sophisticated map tracker (as in food delivery applications).
Flow	Main Success Scenario (Base Flow): 1- Admin opens up the orders pane in the menu. 2- Admin toggles each (unserviced) order and changes its status. 3- Admin updates each order. Extensions(Alternate Flow): 2-a) If order in question is already delivered, then simply refresh the page. This time, the 'serviced' order will be removed from the system.

1.7.4 Use Case 04 - Add Order

Table 1.6: Add Order , Use Case 4

Use Case ID	U04
Name	Add Order
Actor	Rider
Description	A shopkeeper Opens up the menu, checks all available items in stock, and orders them. Each ordered product is stored in the customer's data. Appropriate Controls are available to the customer to regulate the quantity of products composing the order. The rider takes this order and logs this.
Flow	**Main Success Scenario *(Base Flow)***: 1- Rider logs in the system and presses the add order prompt. 2. There are the certain information which the{Rider} has to enter to place the order according to the Requirements given by the **Shopkeeper** which are given below: 2.1- The Rider has to choose the item from the {List} of the items shown in the UI. 2.2-The Rider then has to click on the Add to order prompt. 2.3-New form will be open in which Rider has to given the {Quantity }of the order. 2.4-At the end Rider has to press the {Add to Cart button} to add the order in the cart. 3. After the Order has been added to the cart. The main page will be shown afterwards. **Extensions*(Alternate Flow)***: If at any time, the system fails, then:Rider requests the recovery system to the previous stable state, which includes The Rider logs in to the system and adds a product to the cart which is already present in the cart. Then, the system will add the quantity of the new product to the quantity of the existing product.

1.7.5 Use Case 05 - Servicing Order

Use Case ID	U05
Name	Servicing order
Actor	Administrator (henceforth referred to as Admin)
Description	After a shopkeeper places the order, the order vis a vis all its details come to the admin for processing and servicing. This includes retrieving the requirements from customer orders and dispatching the order (through riders) to customers. The service part ends when the order is received by the customer and acknowledged.
Flow	Base Flow: As the Rider hands over the orders list to the Administrator, the Administrator starts preparing the order for dispatching. 1. The Admin receives order's list from rider and will dispatch the order which has been ordered first. Hence, first comes first served principle will be followed. 2. Admin will select a particular order and then marks its quantity. Then the Administrator will get the particular item in the specific quantity from inventory. 3. After getting loot from the inventory, Administrator will pack the order. 4. After packing all the orders, the orders will be handed over to the Rider and the order status will be updated to dispatched. 5. After the Rider delivers the order, order status will be updated to Delivered. Extensions(Alternate Flow):If at any time the system fails, the Admin requests the recovery of the system to the previous state, which includes, 1(a) At any time, the Admin dispatches the wrong item by mistake. Then the shopkeeper will be able to return the product. 1(b). The product quantity to be dispatched is finished by any reasons, the Admin will restock the product within the time of product delivery. 2. By any unfortunate circumstances, the Rider is not able to deliver the product, product will be delivered with the discount in the payment.

1.7.6 Use Case 06 - Updating Order Status

Table 1.7: Updating Order Status , Use Case 6

Use Case ID	U06
Name	Updating Order Status
Actor	Rider
Description	While dealing with products, admins and riders need to update the status of the product at each step of the process. This can either be done using a plain text messages (as in post applications) or can be used as a sophisticated map tracker (as in food delivery applications).}
Flow	**Main Success Scenario (Base Flow)**: 1- Rider opens up the order tabs in the menu. 2- The orders tabs contains all orders pertaining to that rider thus far. 3- The order in the queue that was toggled thus far, its status will be updated in the form of a combobox. 4- At the delivery of order, its status will be uploaded. **Extensions(Alternate Flow)**: At any point, if the system fails, the rider requests to reload the previous state. In this case, all previous information is restored. 2-a): Sometimes, the orders pane contains the order that the rider has already delivered. In this case, refresh the page. 2-b): If status to be lodged is something other than the items in combobox (other status), a control is provided to the rider so that they can put the appropriate status there. 2-c): If the shopkeeper is unavailable, the rider needs to take back the order to the warehouse. In this case, if the control is toggled, the order status is updated to ”Delivered, NA - Contact Warehouse”. The onus to receive the order is henceforth the responsibility of the shopkeeper.

1.7.7 Use Case 07 - Add Product

Table 1.8: Add Product , Use Case 7

Use Case ID	U07
Name	Add Product
Actor	Admin
Description	Any distribution system needs one particular category of items to function, that is Products. This use case takes care of that fundamental paradigm.
Flow	**Main Success Scenario (Base Flow)**: 1- Admin opens up the add product tab. 2- Admin fills all the identifiers which are: • Name • Price • ID • Category • Is Perishable 3- At toggling of the Add Product pane, the product is added into the system. **Extensions(Alternate Flow)**: At any point, if the system fails, the rider requests to reload the previous state. In this case, all previous information is restored.

1.7.8 Use Case 08 - Update Product

Table 1.9: Update Product , Use Case 8

Use Case ID	U08
Name	Update Product
Actor	Admin
Description	Any distribution system needs one particular category of items to function, that is Products. During operation, various aspects of items need to be changed during runtime. Stock needs to be updated, price needs to be adjusted, tax rate needs to be changed etc. This use case takes care of that fundamental paradigm.
Flow	**Main Success Scenario (Base Flow)**: 1- Admin opens up the Update product tab. 2- Admin updates any of required identifiers which may be: • Price • ID • Category • Is Perishable 3- At toggling of the update Product pane, the product is added into the system. **Extensions(Alternate Flow)**: At any point, if the system fails, the rider requests to reload the previous state. In this case, all previous information is restored. 2-a):If at any point, invalid data is applied: • Appropriate use of validators is applied at this step. • Error, informative popups / icons are employed to assist the admin in operation. Scrolling to the topmost disruptive component is applied also. 3-a): If, any product other than the intended product is updated, the previous state of the product is to be restored. 4-a): If product stock in warehouse is full (overflow condition), the product stock needs to be marked out so that it is dealt with in an orderly manner, i.e, is dispatched first.

1.7.9 Use Case 09 - Billing Payments

Table 1.10: Use Case 9 Billing Payment

Use Case ID	U09
Name	Billing Payments
Actor	Rider
Description	At the time of payments, the shopkeeper needs to pay for the orders they have placed. Since the delivery vehicle for this is the rider, they need to be able to execute payments. The mechanics for this can either be old-school cash payments (either on delivery or in person), advance payments or online. Once a payment is mature, only then a order can be given the go ahead for process.
Flow	Base Flow: Rider will enter the customer's name with the stock of the orders and the total billing payment. At the last the rider will choose any method to confirm the payment. The payments methods are given: 1:Jazz Cash 2:Cash 3:Advance Order payments are matured by the rider through cash or by any external Api's then returned to the admin side loggings. Alternate Flow: At any time, System fails: To support recovery and correct accounting, ensure all transactions sensitive state and Payments can be recovered from any step of the scenario.' 1. Rider restarts System, logs in, and requests recovery of prior state. 2. Invalid term ID then the Rider again starts the system and enter again the further details to ensure the paymet. 3. The payment is given more or less by mistake then the rider will generate a query about refunding the payment and receives the payment signal from the Payment System.

1.7.10 Use Case 10 - Delete Product

Table 1.11: Delete Product , Use Case 10

Use Case ID	U10
Name	Delete Product
Actor	Admin
Description	Any distribution system needs one particular category of items to function, that is Products. During operation, various aspects of items need to be changed during runtime. Stock needs to be updated, price needs to be adjusted, tax rate needs to be changed etc. A product can also be deldeted. This use case takes care of that fundamental paradigm.
Flow	Main Success Scenario (Base Flow): 1- Admin opens up the Delete product tab. 2- Admin presses the delete product tab from that product. 3- At toggling of the delete Product pane, the product is added into the system. Extensions(Alternate Flow): At any point, if the system fails, the rider requests to reload the previous state. In this case, all previous information is restored. 2-a):If at any point, the product is not deleted, a popup is displayed to the user, denoting that the product was not deleted due to the proscribed reason (order of that product yet undelivered, chiefly among other reasons).

1.7.11 Use Case 11 - Logging Reservations

Table 1.12: Logging Reservations , Use Case 11

Use Case ID	U11
Name	Logging Reservations
Actor	Rider
Description	During the mode of operation, the rider may encounter complaints about product. These complaints need to be forwarded to the admin for action.
Flow	Main Success Scenario (Base Flow): 1- Rider opens up the Complaints tab. 2- Rider records the complaints and forwards them to admin. Extensions(Alternate Flow): At any point, if the system fails, the rider requests to reload the previous state. In this case, all previous information is restored.

1.7.12 Use Case 12 - Add route

Table 1.13: Add Route, Use Case 12

Use Case ID	U12
Name	Add Routes
Actor	Rider
Description	When during RunTime a shop is added, it needs to be correctly added within existing network. This is necessary for mainly two reasons: a) Because shortest path from shop 1 (any) to this newly added shop is required, and b) Because daily route of rider needs to be updated.
Flow	Main Success Scenario (Base Flow): 1- When a new shop is added (U02) , at backend using a graph, a new vertex and edge is added. If add shop is executed correctly, this part should be executed seamlessly. Extensions(Alternate Flow): At any point, if the system fails, the rider requests to reload the previous state. In this case, all previous information is restored.

1.7.13 Use Case 13 - Delete Order

Table 1.14: Delete Order, Use Case 13

Use Case ID	U13
Name	Delete Order
Actor	Rider
Description	Sometimes a shopkeeper is not content with an order they wish to delete it, in this case this use case is handy.
Flow	Main Success Scenario (Base Flow): 1- Rider opens the orders pane. 2- Rider selects the target order. 3- Rider presses the Delete order button. 4- Order is deleted from the system. Extensions(Alternate Flow): At any point, if the system fails, the rider requests to reload the previous state. In this case, all previous information is restored. 2-a) If wrong order is deleted, an 'escape' option in form of a button is provided to the user to restore previous order states. 2-b) If order is deleted from the system, but later on a modification is required in the 'deleted order', new order pane is opened. Except the target attribute, all details are filled as they were previously.

1.7.14 Use Case 14 - Authenticate User

Table 1.15: Authenticate User, Use Case 14

Use Case ID	U14
Name	Authenticate User
Actor	Admin , Rider
Description	The entry to any system is the login pane. If the user is entered correctly, then and only then a user can be admitted into the system. What point of entry (Admin) or (Rider) they get depends on their credentials.
Flow	Main Success Scenario (Base Flow): 1- Main Login Screen is opened by the user. 2- User enters their username and password. 3- A user is searched in the backend. 4- If they are found to be a valid user, then they are granted access according to their role. Extensions(Alternate Flow): At any point, if the system fails, the rider requests to reload the previous state. In this case, all previous information is restored. 1-a): If the login screen does not open, the page is reloaded. 2-a): If any input field is empty, 2-a)-1: then an informative popup/icon is displayed to the end user. 2-a)-2: Scrolling to the empty component is also employed to direct the user attention to that component. 3) If a username is found but password doesn't match, 3-a): a message just below password input prompt is displayed. This happens for a total of 3 times. 3-b) If, after three times, the user still can't seem to remember their password, then a clause of 'Forgot Password' is displayed. 3-b)-i) : This will refer them to the security question employed. IF answered correctly, the user's password will be given to them. 3-b)-ii): If security question is remembered wrong, they can't login again.

1.7.15 Use Case 15 - Navigate Route

Table 1.16: Navigate Rider, Use Case 15

Use Case ID	U15
Name	Navigate Route
Actor	Rider
Description	After collecting all the orders from the inventory, the Rider is ready to deliver the orders to their location. To deliver the orders at their right location, the rider needs a roadmap. The Rider follows the roadmap and delivers orders before the specified time.
Flow	**Base Flow**: As all the orders are packed for the specific route, the Rider of that route picks the orders for the inventory and is ready to deliver. 1. The Rider logs in to system.then check in to the orders collection center and if the orders are packed completely, Rider picks at the start of the day. 2. The Rider will start delivering the orders one by one, by the following process: 2 (a). Rider Selects one order from the pending orders list and will click Start Delivering button on the screen. 2 (b). After clicking the button, map will display on the screen will give route to the delivery address of that particular order. 2 (c). The Rider will navigate through the route to reach to location earlier than expected. 3. As the Rider reaches the location of order delivery, the map will disappear and and an End Order Button will appear. 4. As the Rider clicks the End Order Button previous order will be shifted to the fulfilled orders sections and a new order route will be given to the Rider. **Extensions(Alternate Flow)**:If at any time the system fails, the Admin requests the recovery of the system to the previous state, which includes, 1(a) The Rider delivers the wrong order to shopkeeper then the shopkeeper will be able to refund the order. 1(b). In case of absence of Rider of a route, the orders delivery will be either postponed and orders will be delivered with discounted price or the orders will be shifted to another Rider. 2. During Navigation, if the Rider enters any irrelevant path. Then the map will be updated to the Shortest-Path according to the Rider's current location. 3. In case Rider does not click End Order Button after delivering an order. Then the next order delivery roadmap will be displayed after completing 1 min 30 sec.

1.7.16 Use Case 16 - View Sales Dashboard

Table 1.17: View Sales Dashboard, Use Case 16

Use Case ID	U16
Name	View Sales Dashboard
Actor	Admin
Description	Any business owner needs to realistically view the statistics of sales trends in a given period of time. This needs to be done in form of figures (numeric).
Flow	Main Success Scenario (Base Flow): 1- Main Login Screen is opened by the user. 2- User opens up Sales Dashboard. Extensions(Alternate Flow): At any point, if the system fails, the rider requests to reload the previous state. In this case, all previous information is restored. 1-a): If the dashboard does not open, the page is reloaded. 2-a): If dashboard contains seemingly incorrect data: 2-a)-1: If it contains negative data (indicating returns/damages), an informative underlying text is displayed. 2-b) If the dashboard does not contain any data: 2-b-1: Reload the page. 2-b-2: If it still doesn't contain, it might be start of a new timespan(week,month etc.) If this is the case, show an underlying text info.

1.7.17 Use Case 17 - Check Fuel Logs

Table 1.18: Check Fuel Logs, Use Case 17

Use Case ID	U17
Name	Check Fuel Logs
Actor	Rider
Description	After collecting all the orders from the inventory, the Rider is ready to deliver the orders to their location. To deliver all the orders successfully, the order needs to refill the petrol tank on daily basis. An amount will be given to the rider on weekly basis which rider will use to refill the petrol tank.
Flow	**Main Success Scenario(Base Flow)**: As all the orders are packed for the specific route, the Rider of that route picks the orders for the inventory and is ready to deliver. 1. The Rider logs in to system. Then check in to the orders collection center and if the orders are packed completely, Rider picks at the start of the day. 2. The Rider will be given Rs. 2000/- on weekly basis to assist the rider ride through the routes. 3. Rider will not be allowed to use over Rs.499/- to refill the tank in day. 4. The Rider will refill the tank using unique amount everyday e.g. if Rider has refilled the tank using Rs. 300/- , he/she cannot refill using Rs. 300/- on any other day of the week. After a week, fuel logs will reset. 5. Rider logs in to system and is displayed Rider Dashboard. After clicking check fuel logs from the dashboard the rider will click Refill button. After clicking button Rider will enter the amount he/she want to refill. **Extensions(Alternate Flow))**:If at any time the system fails, the Admin requests the recovery of the system to the previous state, which includes, 1(a) In case the rider selects amount greater than Rs. 499/- for refilling in a single day, the rider will be given a warning message to use less than Rs. 500/- for refilling in a single day. 1(b). In case the rider selects an amount which he\she already has used in that same week, he\she will be given a unique message that the amount is not unique.

1.7.18 Use Case 18 - View Order History

Table 1.19: View Order History, Use Case 18

Use Case ID	U18
Name	View Order History
Actor	Admin
Description	Previous order trends need to be monitored time by time in order to guage what sales pattern is currently in flow with the shopkeepers. For this purpose, records about Product and pays need to be periodically maintained.
Flow	**Main Success Scenario (Base Flow)**: 1- Admin opens up the previous orders history pane from the menu. 2- Admin has a view of 4 products ordered (rest of them will be in a csv file) 3- Admin can view and download the csv in any directory of their choice. **Extensions(Alternate Flow)**: At any point, if the system fails, the rider requests to reload the previous state. In this case, all previous information is restored. 2-a): If csv file contains seemingly incorrect data: 2-a)-1: If it contains negative data (indicating returns/damages),an extra column in csv denoting status is filled with the status of order. 2-b) If the csv does not contain any data: Fill the C2 column of csv with the message 'Data not available due to new operations'.

1.7.19 Use Case 19 - View Rider

Table 1.20: View Rider, Use Case 19

Use Case ID	U19
Name	View Rider
Actor	Administrator
Description	The Administrator as a lead is able to control and analyze the distribution system. The Administrator will be able to add riders. Hence, he\she will be able to view riders data. As a result, Admin has the control to manage riders.
Flow	**Main Success Scenario(Base Flow)**: The Admin as a system manager has the control to manage rides. Therefore, has the command to do the following tasks: 1. The Admin logs in to system. Then Admin Dashboard will appear to Admin. To view the Riders Admin will select Manage Riders Button from the menu. 2. After clicking the button the Admin will be directed to Manage Riders page. 3. Here Riders will be displayed in a list view. Every Rider in the list will have 2 buttons namely View Single Rider and Delete Rider. 4. To view single rider click on the view single rider button and the Admin will be directed to Rider Data Page. **Extensions(Alternate Flow)**:If at any time the system fails, the Admin requests the recovery of the system to the previous state, which includes, 1(a) The Admin logs in to system. In case the device shuts down or application crashes, the system will store its current state and Admin will be directed to the same page where the system was standing before the shutdown.

1.7.20 Use Case 20 - Delete Rider

Table 1.21: Use Case 20 Deleting Rider

Use Case ID	U20
Name	Delete Rider
Actor	Admin
Description	The riders which are already present in the management team are sometimes need to be shifted to other positions or to be fired from the management team. Thus, by using the given prompt it is possible to delete a rider from the team to upgrade the system according to the requirements. If there is any need to update the riders then this is also be used by the admin to perform the tasks.
Flow	**Main Success Scenario (Base Flow)**: 1. The Admin first logs in to the system using his credentials and after confirmation admin clicks on the delete rider prompt to enter to the system to delete a specific rider according to the inquiry against the rider if any. 2. There will be a list of riders which are registered already into the system in which' the information of a rider is given as below: a. Rider's name b. Rider's ID c. Rider's Present Position 3. After the rider has been selected the admin at last has to pressed the delete prompt for the confirmation to delete the selected rider and an email will be generatred which will further sent to the rider for his acknowledgment. **Extensions (Alternate Flow)**: 1. If there is any problem in the admin's credentials that admin can not logs in to the system then the admin will again first register then again logs in to the system to perform his duty. 2. If admin mistakenly delete any other rider then admin first logs in to the add rider to registered the deleted rider again. 3. If there is no rider according to the info which has to be deleted then there will be two tasks performed by the admin which are given below: a. Admin refresh the system if there is any glitch. b. Admin will take a completer procedure to registered the rider first then delete it from the system. 4. If the email is not generated and sent to the rider Admin will be interact with the developing team to re-consider the system again.

1.7.21 Use Case 21 - Update Rider

Table 1.22: Update Rider, Use Case 21

Use Case ID	U21
Name	Update Rider
Actor	Rider
Description	Rider needs to update their details. This use case comes in handy in that aspect.
Flow	**Main Success Scenario (Base Flow)**: 1- Rider opens up the 'Update your details' pane from the menu. 2- A page in this regards opens up. 3- The page has textboxes containing the details. Only the text box containing Rider CNIC will be blocked from modification. 4- Rider can modify their details as they wish. **Extensions(Alternate Flow)**: At any point, if the system fails, the rider requests to reload the previous state. In this case, all previous information is restored. This goes without saying that the rider need not fill the information again. 3-a): If the rider enters incorrect phone number: 3-a-s): Appropriate use of validators will be applied in this case *(length of phone num ber must be 11 digits and starts with 03 and must not contain any letter).* 2-b) If password is edited, its length must be 8 digits. 4-a): If, a rider detail isn't modified after the process, an 'escape' facility in form of 'Previous Changes' is applied. It will show changes *(not completely, just in required places)* which the user can then synchronize. 4-b): If a field is left empty, then an informative icon/popup is displayed so that the end user can directly go to that field and do that.

1.7.22 Use Case 22 - Send Emails

Table 1.23: Send Emails, Use Case 22

Use Case ID	U22
Name	Send Emails
Actor	Admin
Description	After a task is completed, the administrator is responsible for contacting the customer and sending a confirmation email, such as "Your order has been placed successfully!" or "Your order has been delivered!".
Flow	**Main Success Scenario (Base Flow)**: 1- When an order is successfully placed in the system (U04), an email is delivered to the customer having text "Your order has been placed successfully". 2- When the rider succesfully delivers orders, an email to this effect will be sent. *See Emails Section for more details.* **Extensions(Alternate Flow)**: 1- If an order is not placed due to payment defeciency or stock reasons, an email will be sent to this effect. 2- If an order is dispatched, but the shopkeeper was not present, an email will be sent to this regard. This will denote the shopkeeper must now recieve the order personally from the warehouse.

1.7.23 Use Case 23 - Save Data

Table 1.24: Save Data, Use Case 23

Use Case ID	U23
Name	Save Data
Actor	Admin, Rider
Description	Data Storage is an essential part of any application flow. This case deals with this use case.
Flow	**Main Success Scenario (Base Flow)**: 1- At every step of application flow, data is stored. 2- Data Structure and file used for the format depends on use case involved. *See Data Storage Section for more details.*

1.7.24 Use Case 24 - Update Shop

Table 1.25: Update Shop, Use Case 24

Use case ID	U24
Name	Update Shop
Actor	Rider
Description	At any time of operation, the shop details need to be updated. This may include, but is not limited to, shopkeeper contact number is changed, shop mode is changed etc. This use case deals with this paradigm.
Flow	Main Success Scenario(Base Flow): 1- Rider opens up update shop pane. 2- A page opens up in this regard. 3- Rider updates required shop details in this regard. Extensions (Alternate Flow): At any point, if the system fails, the rider requests to reload the previous state. In this case, all previous information is restored. 3-a) If any field is left empty, an informative pane is displayed in this regard. Appropriate measures such as scrolling to the vacant component are also employed. 3-b) If any invalid data is entered, appropriate use of validators is employed. For purposes of data security, updating critical features such as CNIC of shopkeeper are prohibited. 3-c) If any wrong shop data is modified, reverting back to previous state is to be made possible.

1.7.25 Use Case 25 - View Shop

Table 1.26: View Shop, Use Case 25

Use case ID	U25
Name	View Shop
Actor	Admin
Description	At any time of operation, the shop details need to be viewed. This may include, but is not limited to, shopkeeper contact number is changed, shop mode etc. This use case deals with this paradigm.
Flow	Main Success Scenario(Base Flow): 1- Admin opens up view shop pane. 2- A page opens up in this regard. 3- Shop details appear in a list format. 4- Admin views required shop details in this regard. Extensions (Alternate Flow): At any point, if the system fails, the rider requests to reload the previous state. In this case, all previous information is restored.

1.7.26 Use Case 26 - View Product

Table 1.27: View Product, Use Case 26

Use case ID	U26
Name	View Product
Actor	Admin , Rider
Description	At any time of operation, the product details need to be viewed. This use case deals with this paradigm.
Flow	Main Success Scenario(Base Flow): 1- User navigates to and opens up view product pane. 2- Shop details appear in a list format. 3- User views required shop details in this regard. Extensions (Alternate Flow): At any point, if the system fails, the rider requests to reload the previous state. In this case, all previous information is restored.

1.7.27 Use Case 27 - Delete Product

Table 1.28: Use Case 27, Delete Product

Use Case ID	U27
Name	Delete Product
Actor	Administrator
Description	The Administrator as a lead is able to control and analyze the distribution system. The Administrator will be able to add products. Hence, he\she will be able to view products data. As a result, Admin has the control to manage inventory.
Flow	**Main Success Scenario(Base Flow):** The Admin as a system manager has the control to manage products. Therefore, has the command to do the following tasks: 1. The Admin logs in to system. Then Admin Dashboard will appear to Admin. To delete the products Admin will select Inventory Button from the menu. 2. After clicking the button the Admin will be directed to Inventory page. 3. Here products will be displayed in a list view. Every product in the list will have 3 buttons namely View Single Product, Edit Product and Delete Product. 4. To delete a single product click on the delete button and the product will be deleted from the inventory. **Extensions(Alternate Flow)**:If at any time the system fails, the Admin requests the recovery of the system to the previous state, which includes, 1 (a) The Admin logs in to system. In case the device shuts down or application crashes, the system will store its current state and Admin will be directed to the same page where the system was standing before the shutdown. 1 (b). In case the Admin deletes a specific product from the inventory but it still displays at the Riders Side Interface. If the Rider tries to add that product to the cart, a Warning message will display saying ”This Product is no more Available”.

1.7.28 Use Case 28 - Analyze Reports

Table 1.29: Use Case 28 Analyze Reports

Use Case ID	U28
Name	Analyze Reports
Actor	Admin
Description	Reports are basically the records which convey the business activities and the financial reports on the monthly basis. Admins on monthly basis considered the reports for knowing what and which percent their systems and workers gave them output and analyze reports and records completely for the betterment of the their companies. Admins will analyze the reports and considered the profits and the total revenue on the daily and the monthly basis.
Flow	**Base Flow**: 1. If any admin has to analyze the report then firstly admin has to logs in to the system and presses the Reports section to analyze it. 2. There will be the graphs and the boxes which shows the following information: 2.1 Total Monthly Profit percentage 2.2 Total Monthly Customers rate 2.3 Total Revenue Generated on Monthly Basis 3. There will be a total Traffic graph also shown in the dashboard so the admin can easily take an estimate about the total social network of people with their Management system. **Alternate Flow** : If there is any deficiency in profit percentage and monthly revenue then there will be a query generated by the admin which will be responded and the percentage will re-calculated.

1.7.29 Use Case 29 - Delete Shop

Table 1.30: Delete Shop, Use Case 29

Use Case ID	U29
Name	Delete Shop
Actor	Rider
Description	The Rider as a co-lead is able to control the distribution system. The Rider will be able to add shops. Hence, he\she will be able to delete shops. As a result, Rider has the control to manage shops.
Flow	**Main Success Scenario(Base Flow)**: The Rider as a system manager has the control to manage shops. Therefore, has the command to do the following tasks: 1. The Rider logs in to system. Then Rider Dashboard will appear to Rider. To delete the shops Rider will select View Shops Button from the menu. 2. After clicking the button the Rider will be directed to Shops page. 3. Here Shops will be displayed in a list view. Every shop in the list will have 3 buttons namely View Single Shop, Edit Shop and Delete Shop. 4. To delete a single shop click on the delete button and the shop will be deleted from the shops list. **Extensions(Alternate Flow)**:If at any time the system fails, the Admin requests the recovery of the system to the previous state, which includes, 1 (a) The Rider logs in to system. In case the device shuts down or application crashes, the system will store its current state and Admin will be directed to the same page where the system was standing before the shutdown. 1 (b). In case the Rider deletes a specific shop from the shops-list but it still displays at the Admin Side Interface. If the Admin will try to do anything with that particular shop a Warning message will display saying ”This Shop is no more On Our Customers List”. 2. After Deleting a shop the Shortest Path Property will hold i.e. every route will have minimum distance as compared to other routes following the same destination.

1.8 User Interfaces

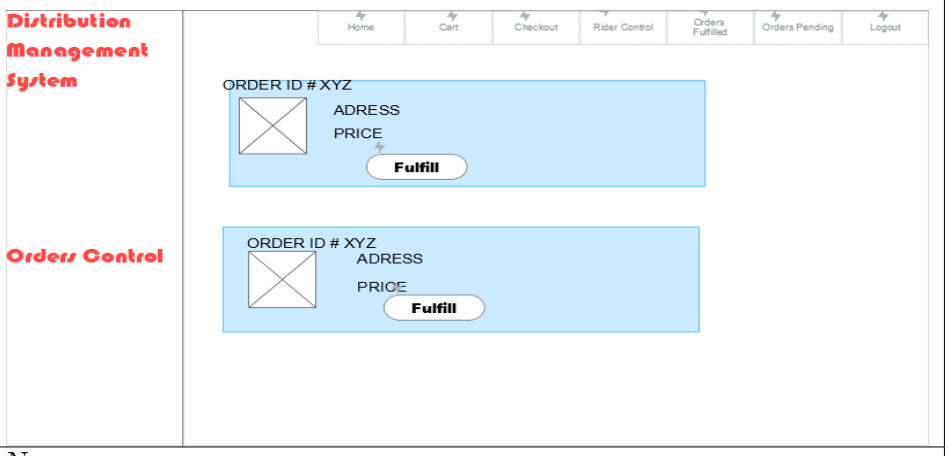
1.8.1 Interface 01

Table 1.31: Add Order Interface, I01

Interface ID	I01
Interface Name	Add Order
Linked Use Case ID	U04
UI Screen (Justinmind)	
Validators	Illegal Data Entries on quantity

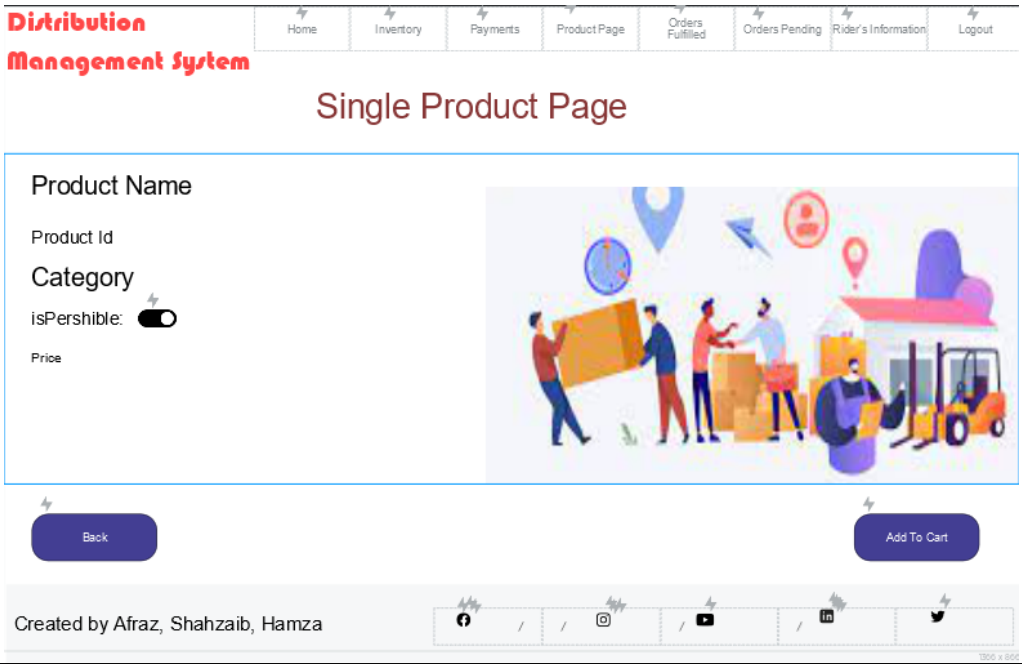
1.8.2 Interface 02

Table 1.32: Service Order Interface, I02

Interface ID	I02
Interface Name	Service Order
Linked Use Case ID	U05
UI Screen (Justinmind)	
Validators	None

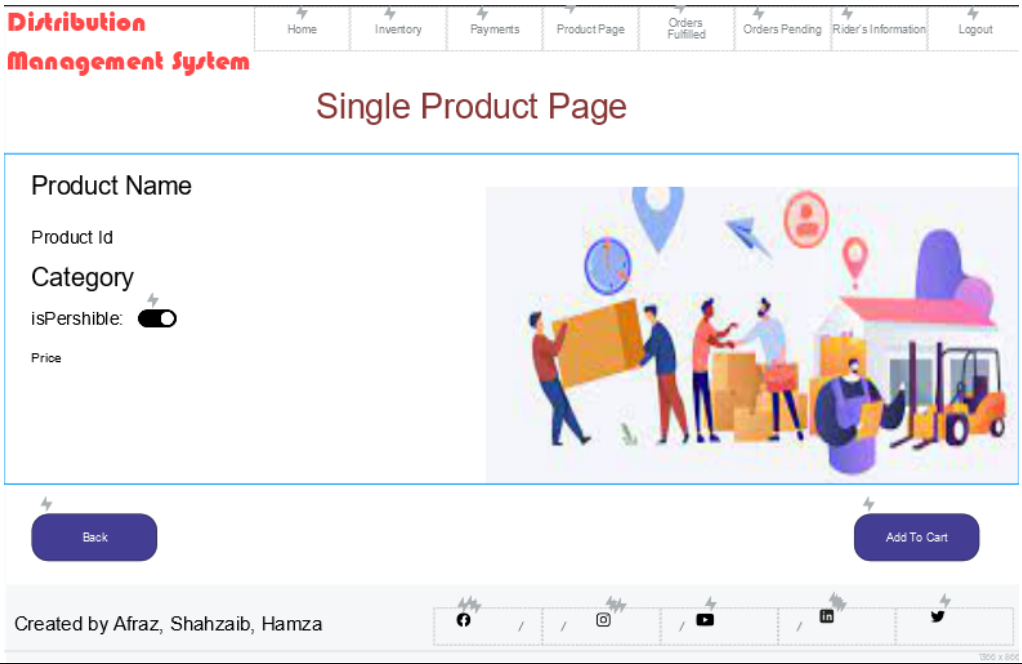
1.8.3 Interface 03

Table 1.33: Add Product Interface, I03

Interface ID	I03
Interface Name	Add Product
Linked Use Case ID	U07
UI Screen (Justinmind)	
Validators	Non empty fields, price in float, tax ratio in float

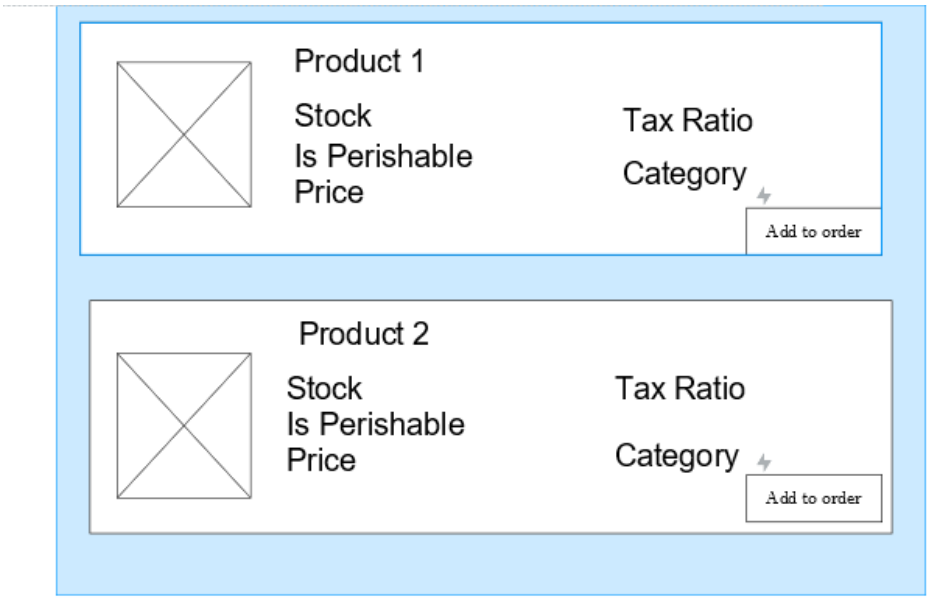
1.8.4 Interface 04

Table 1.34: Update Product Interface, I04

Interface ID	I04
Interface Name	Update Product
Linked Use Case ID	U07
UI Screen (Justinmind)	
Validators	Non empty fields, price in float, tax ratio in float, stock attribute inserted

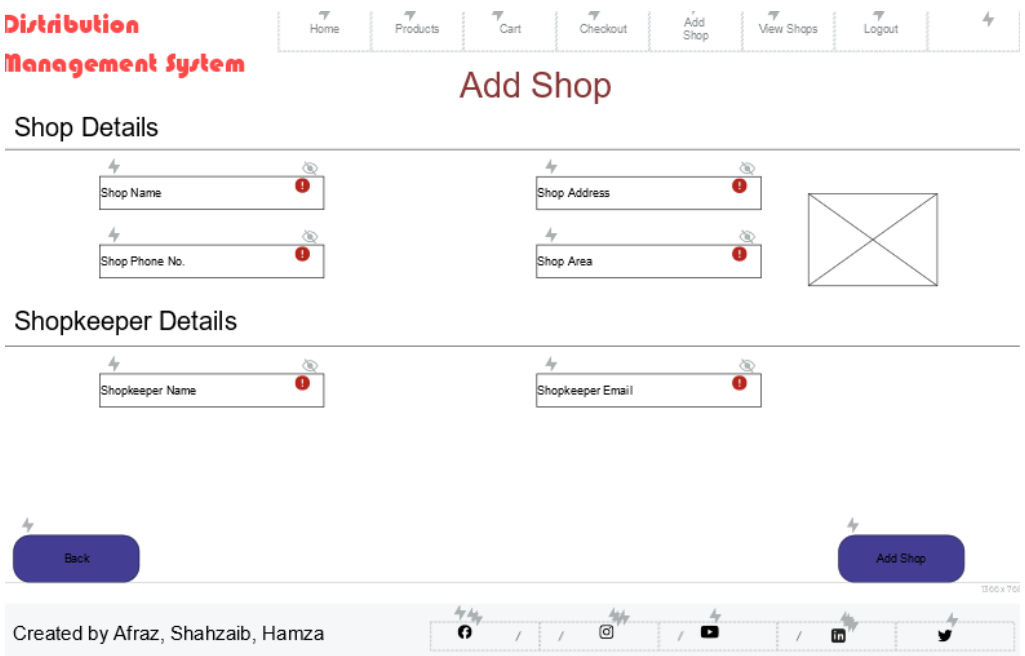
1.8.5 Interface 05

Table 1.35: View Product Interface, I05

Interface ID	I05
Interface Name	View Product
Linked Use Case ID	U09
UI Screen (Justinmind)	
Validators	None

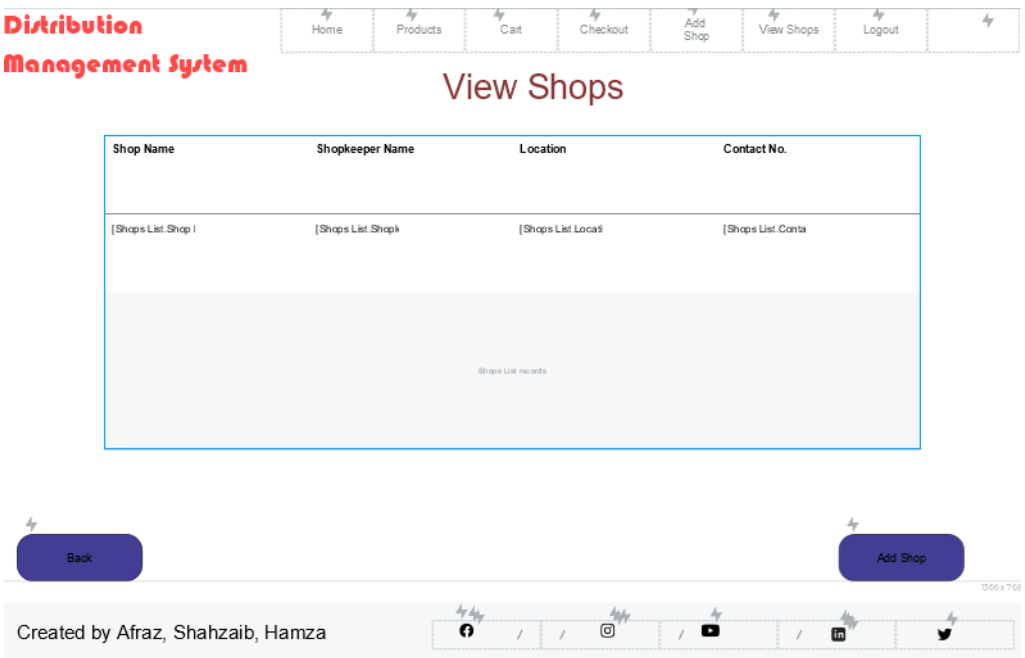
1.8.6 Interface 06

Table 1.36: Add Shop Interface, I06

Interface ID	IO6
Interface Name	Add Shop
Linked Use Case ID	U02
UI Screen (Justinmind)	
Validators	Phone Number length is 11, no letters , starts with 03 Password length is 8 atleast

1.8.7 Interface 07

Table 1.37: View/Delete Shop Interface, I07

Interface ID	IO7
Interface Name	View/Delete Shop
Linked Use Case ID	U25/29
UI Screen (Justinmind)	
Validators	None

1.8.8 Interface 08

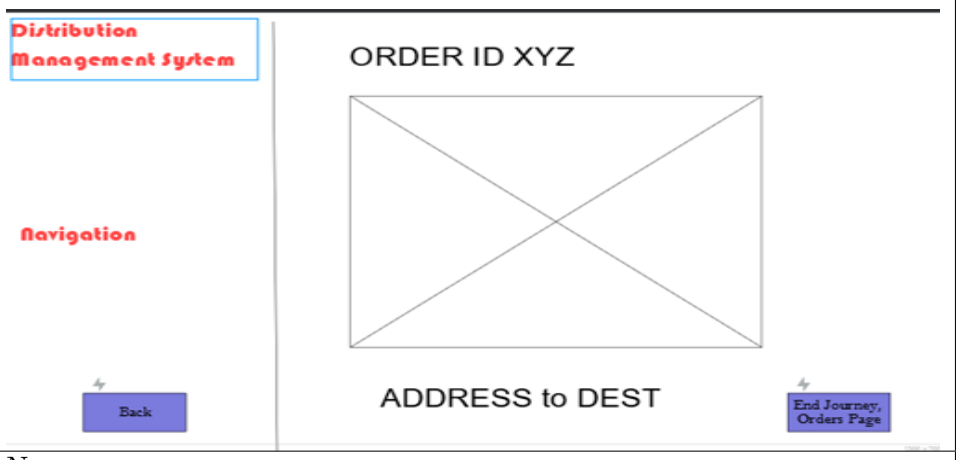
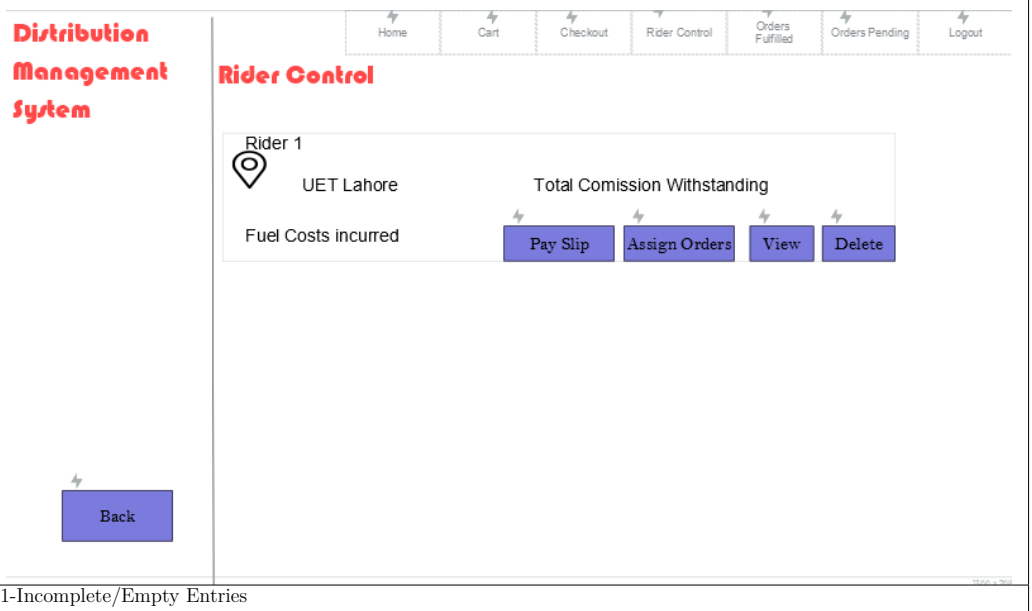
Interface ID	I08
Name	Navigation
Linked Use Case	U10
UI Screen (JustInMind)	
Validators	None

Table 1.38: Interface 08 of the project

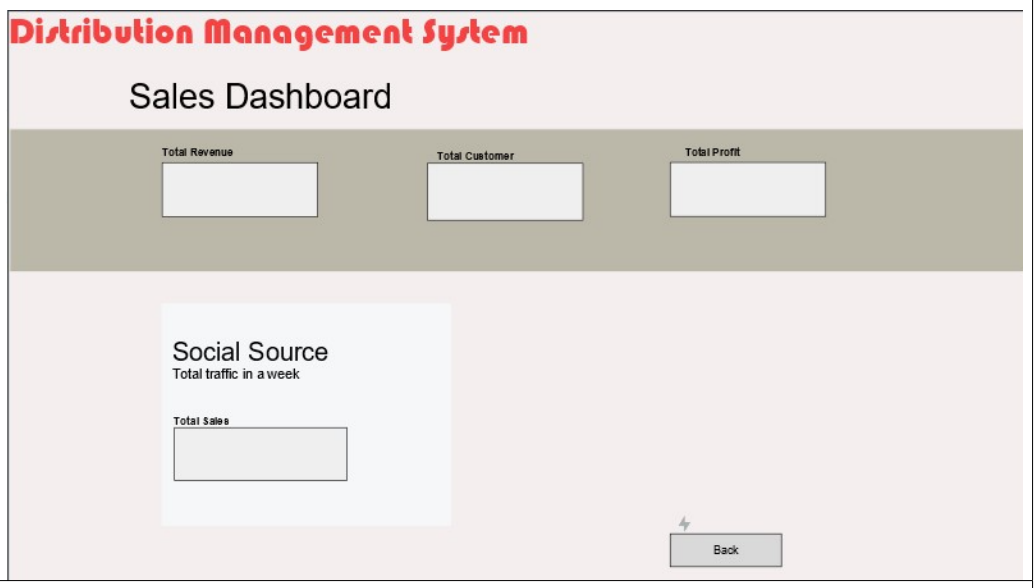
1.8.9 Interface 09

Table 1.39: Rider Fuel Costs Interface, I09

Interface ID	I09
Interface Name	Rider Fuel Costs
Linked Use Case ID	U11
UI Screen (Justinmind)	
Validators	1-Incomplete/Empty Entries 2-Phone number must be of 11 digits, starts with 03 and must not contain any letter

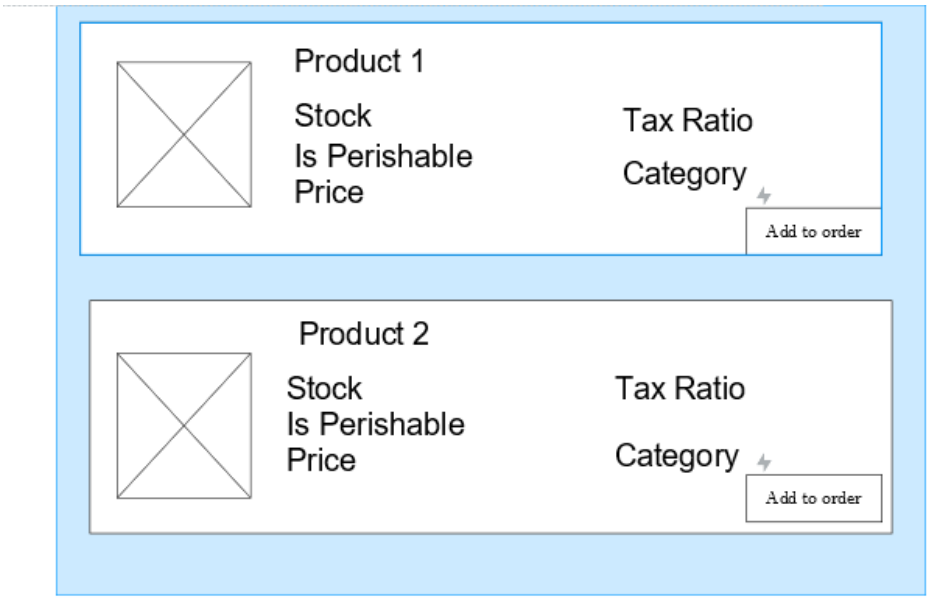
1.8.10 Interface 10

Table 1.40: Sales Dashboard Interface, I10

Interface ID	I10
Interface Name	Sales Dashboard
Linked Use Case ID	U10
UI Screen (Justinmind)	
Validators	none

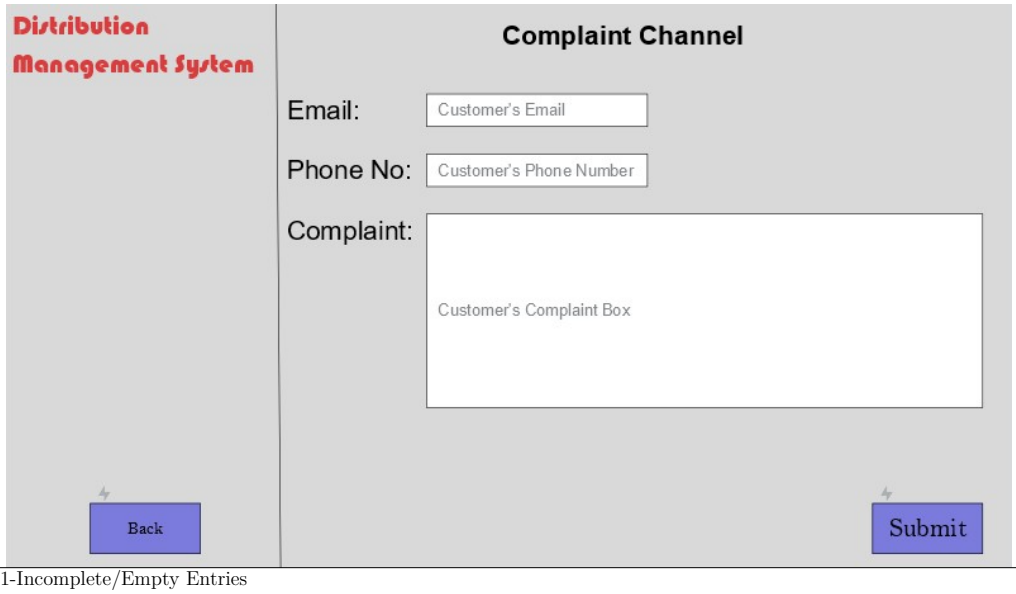
1.8.11 Interface 12

Table 1.41: Delete Product Interface, I12

Interface ID	I12
Interface Name	Delete Product
Linked Use Case ID	U09
UI Screen (Justinmind)	
Validators	None

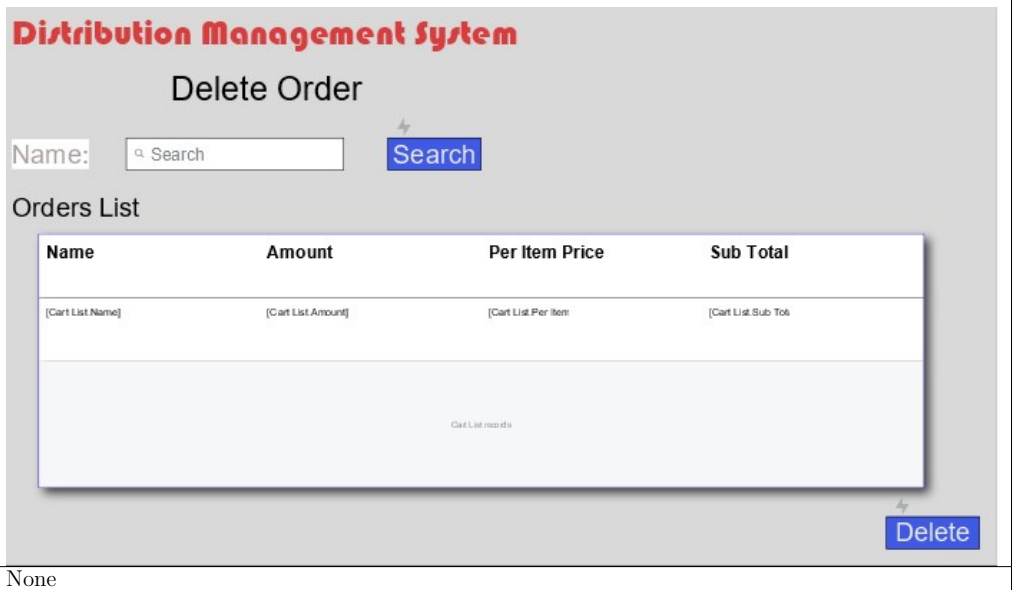
1.8.12 Interface 13

Table 1.42: Complaints Channel Interface, I13

Interface ID	I13
Interface Name	Complaints Channel
Linked Use Case ID	U11
UI Screen (Justinmind)	
Validators	1-Incomplete/Empty Entries 2-Phone number must be of 11 digits, starts with 03 and must not contain any letter

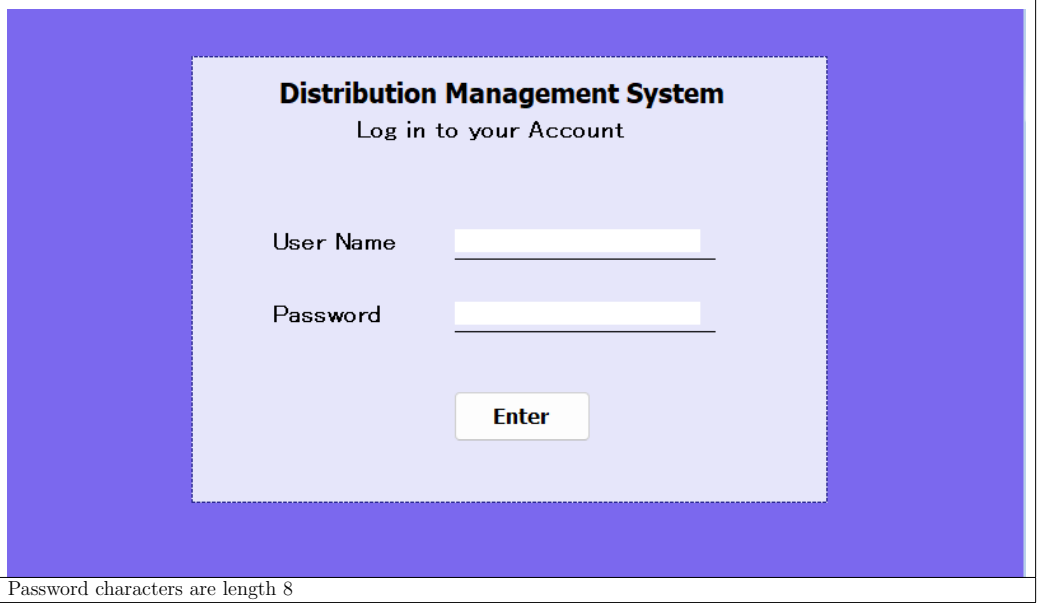
1.8.13 Interface 14

Table 1.43: Delete Product Interface, I14

Interface ID	I14
Interface Name	Delete Order
Linked Use Case ID	U15
UI Screen (Justinmind)	
Validators	None

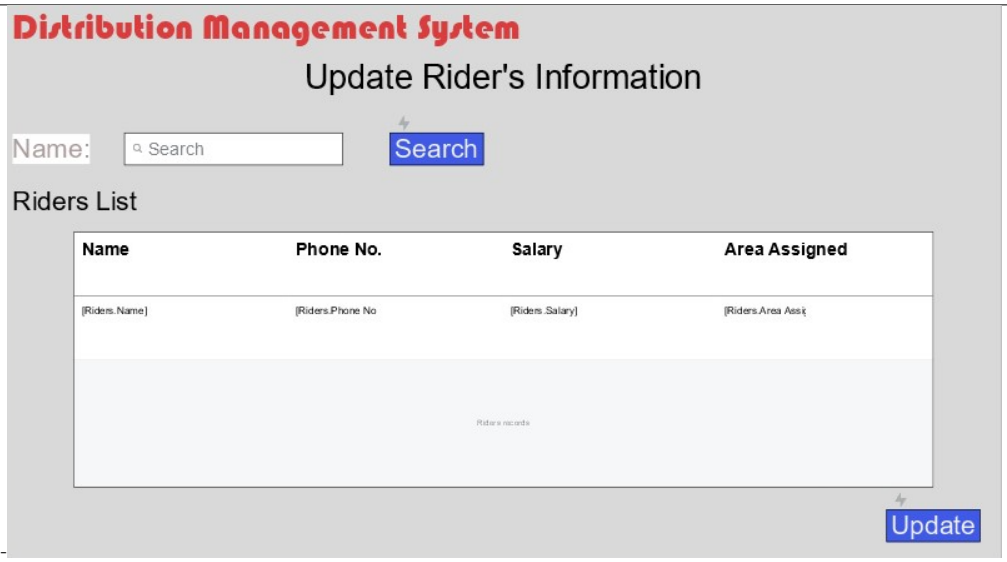
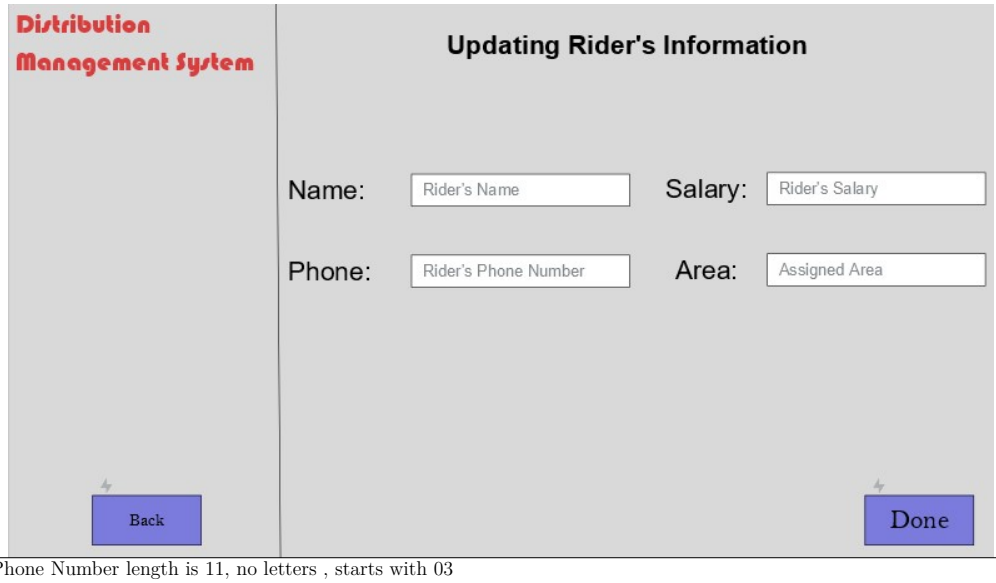
1.8.14 Interface 15

Table 1.44: Login Interface, I15

Interface ID	I15
Interface Name	Login
Linked Use Case ID	U14
UI Screen (Justinmind)	
Validators	Password characters are length 8

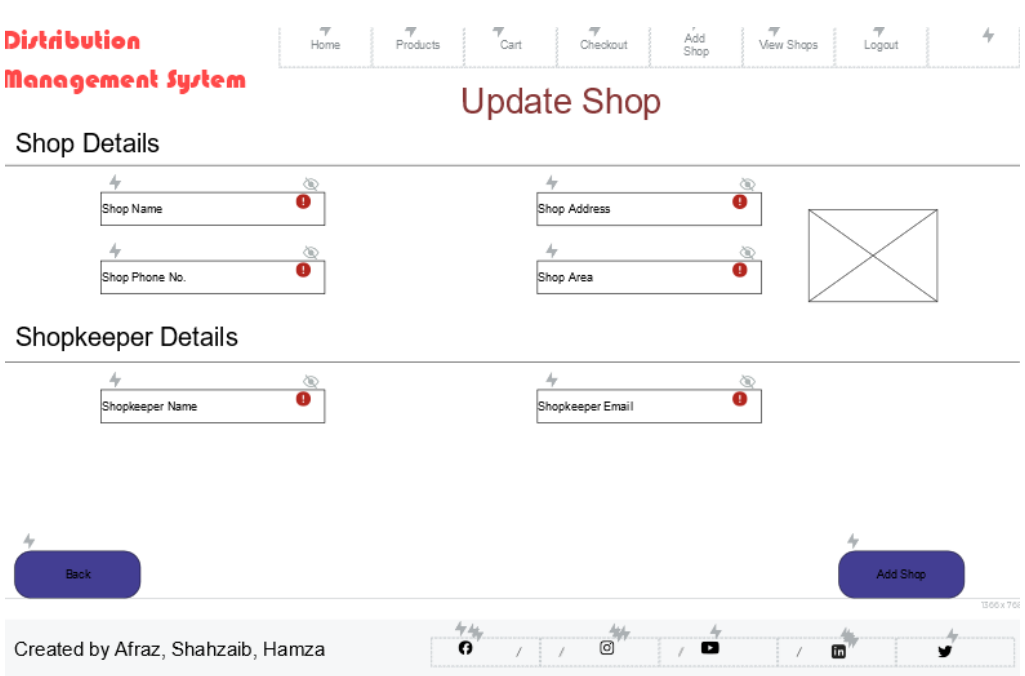
1.8.15 Interface 17

Table 1.45: Update Rider Interface, I17

Interface ID	I17
Interface Name	Update Rider
Linked Use Case ID	U21
UI Screen (Justinmind)	 
Validators	Phone Number length is 11, no letters , starts with 03 Password length is 8 atleast

1.8.16 Interface 18

Table 1.46: Update Shop Interface, I18

Interface ID	I18
Interface Name	Update Shop
Linked Use Case ID	U024
UI Screen (Justinmind)	
Validators	Phone Number length is 11, no letters , starts with 03 Password length is 8 atleast

1.9 Classes

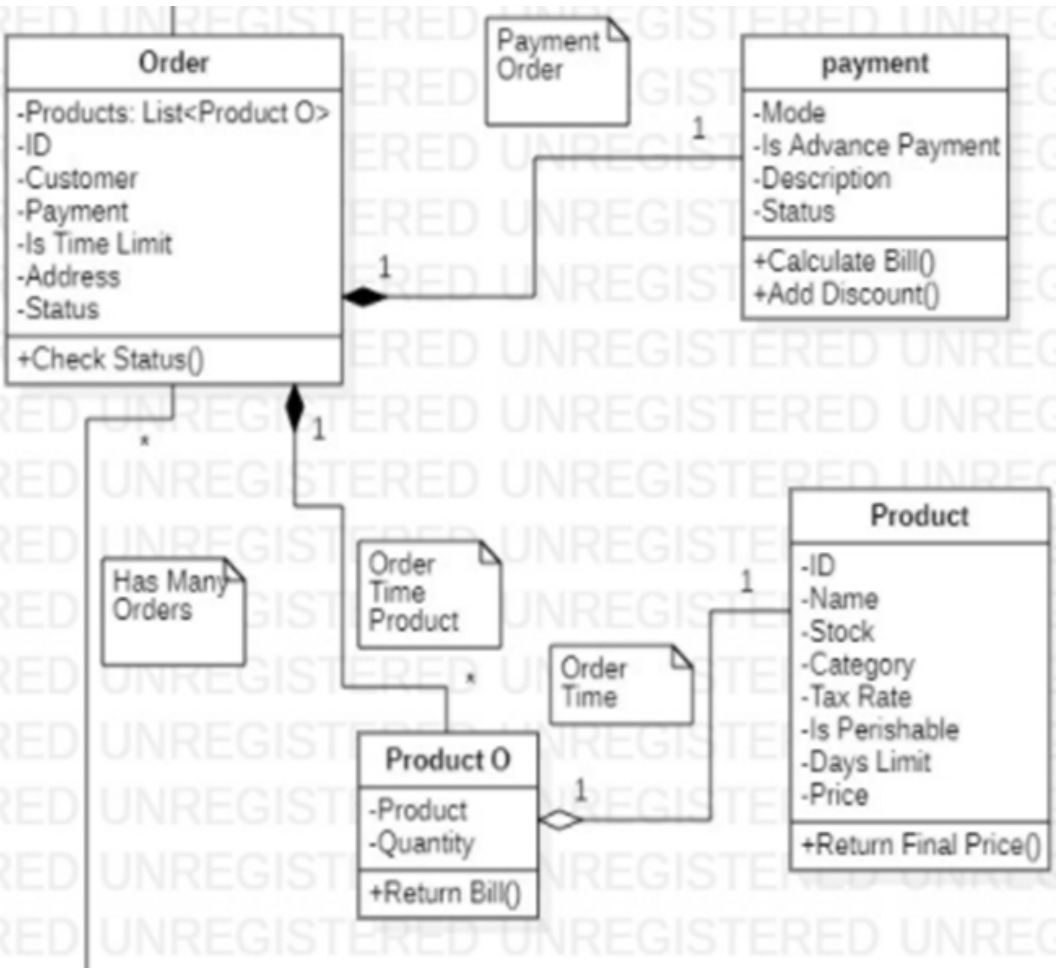
Following is the summary of classes in our project proposal.

Table 1.47: Class Summary

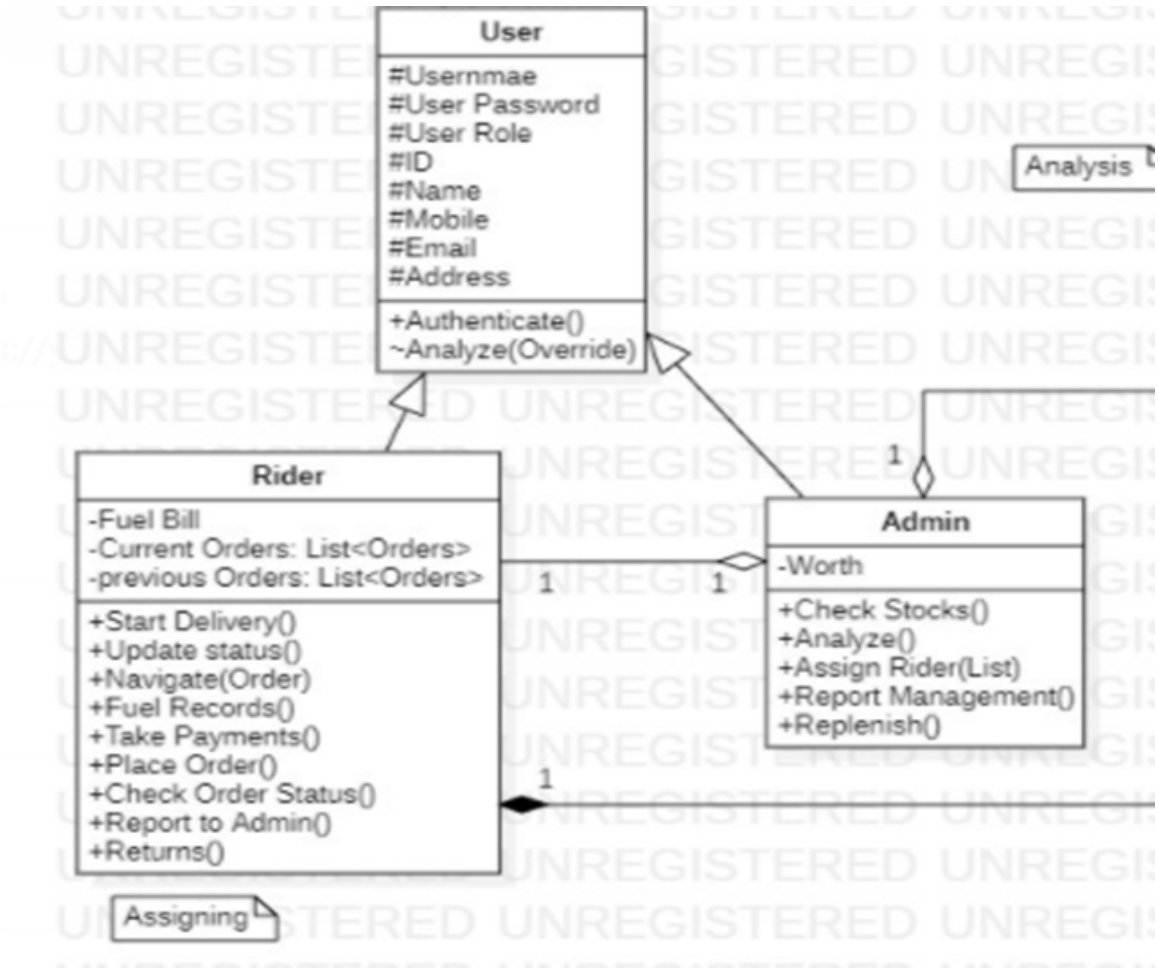
Class Name	Software/Domain	Is Abstract (Yes/No)	Is Singleton (Yes/No)	Is the class will has parametrized Constructor (yes/No)
User	Real World	NO	No	Yes
Payment	Software		Yes	
Rider	Real World			
Admin	Real World			
Product	Real World			
LineItem	Business			
Shopkeeper	Real World			
Order	Business			
Directions	Software			
Shop	Business			

1.10 Object Oriented Features

1.10.1 Composition



1.10.2 Inheritance



1.10.3 Multiple Inheritance

Multiple inheritance is not implemented in our project.

1.10.4 Multi level Inheritance

Multi level inheritance is not implemented *per se* in our project. All instances of inheritance are single level only. However, multiple inheritance using an interface (*or implementation of an interface*) is given. See Figure 1.1 on next page.

1.10.5 Polymorphism

Polymorphism in the project is implemented through virtual override method in User Class. See Figure 1.2 on next page.

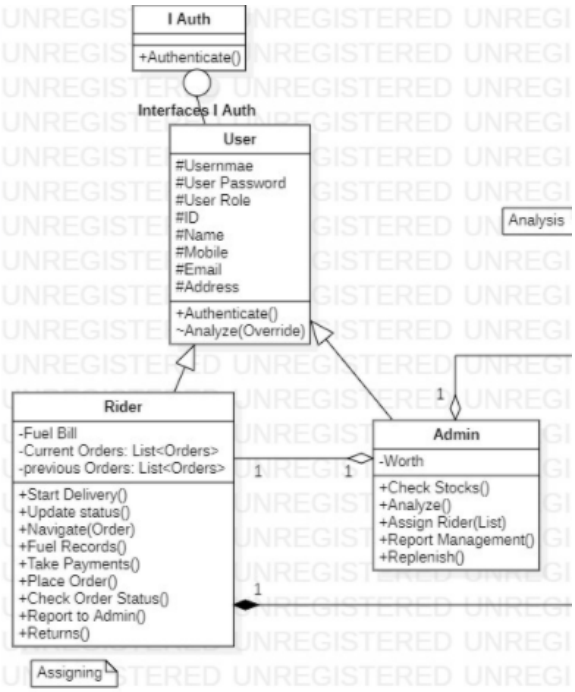


Figure 1.1: Multi level Inheritance

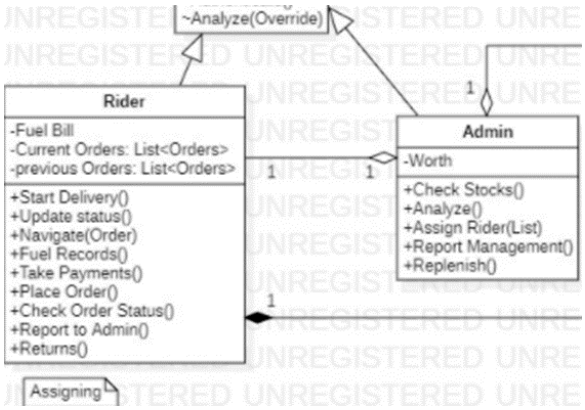
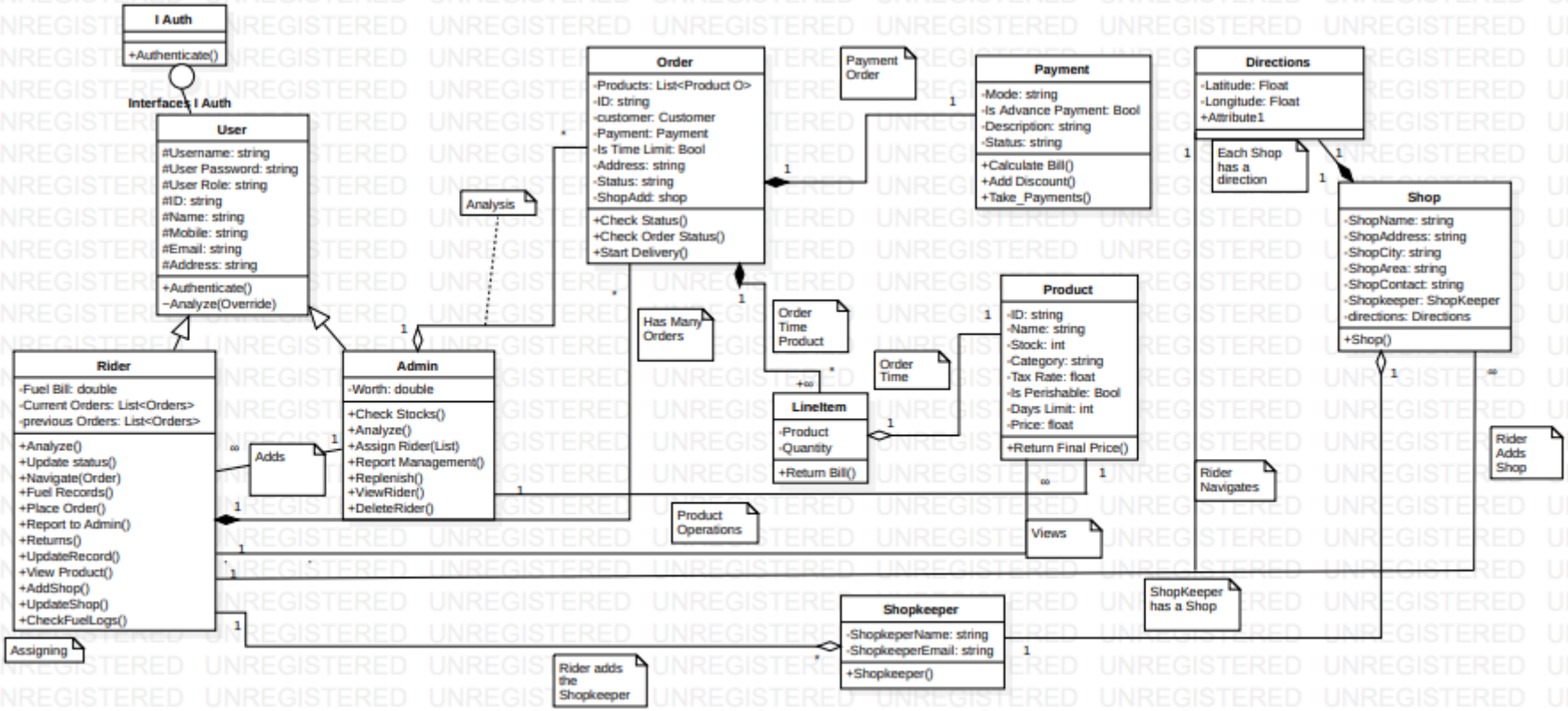


Figure 1.2: Polymorphism

1.10.6 Detailed Object Oriented Design

This section has the detailed, Business Logic UML diagram made in StarUML tool. User is the parent class of Rider and Admin. Order composes inside rider.


I Auth
+Authenticate()
Interface: I Auth
User
#Username: string
#User Password: string
#User Role: string
#ID: string
#Name: string
#Mobile: string
#Email: string
#Address: string
+Authenticate()
−Analyze(Override)
Rider
-Fuel Bill: double
-Current Orders: List<Orders>
-previous Orders: List<Orders>
+Analyze()
+Update status()
+Navigate(Order)
+Fuel Records()
+Place Order()
+Report to Admin()
+Returns()
+UpdateRecord()
+View Product()
+AddShop()
+UpdateShop()
+CheckFuelLogs()
Admin
-Worth: double
+Check Stocks()
+Analyze()
+Assign Rider(List)
+Report Management()
+Replenish()
+ViewRider()
+DeleteRider()
Order
-Products: List<Product O>
-ID: string
-customer: Customer
-Payment: Payment
-Is Time Limit: Bool
-Address: string
-Status: string
-ShopAdd: shop
+Check Status()
+Check Order Status()
+Start Delivery()
Payment
-Mode: string
-Is Advance Payment: Bool
-Description: string
-Status: string
+Calculate Bill()
+Add Discount()
+Take_Payments()
Directions
-Latitude: Float
-Longitude: Float
+Attribute1
Shop
-ShopName: string
-ShopAddress: string
-ShopCity: string
-ShopArea: string
-ShopContact: string
-Shopkeeper: ShopKeeper
-directions: Directions
+Shop()
Product
-ID: string
-Name: string
-Stock: int
-Category: string
-Tax Rate: float
-Is Perishable: Bool
-Days Limit: int
-Price: float
+Return Final Price()
LineItem
-Product
-Quantity
+Return Bill()
Shopkeeper
-ShopkeperName: string
-ShopkeeperEmail: string
+Shopkeeper()
Analysis
Adds
Assigning
Has Many Orders
Order Time Product
Order Time
Product Operations
Views
Rider Navigates
Rider Adds Shop
ShopKeeper has a Shop
Rider adds the Shopkeeper
Payment Order
Each Shop has a direction

1.11 Data Structures and Usage

Table 1.48: Data Structures in project linked with use cases

Use Case Id	Data Structures Used	Justification
U01,U19,U20,U21	Link List	To store rider Data, their is no specific order (LIFO,FIFO), that needs to be applied. The main reason for link list preference to BST is that rider names may be same. This hampers BST search operation and adds to complexity of the search operation in BST.
U02, U24, U25, U29	Graph	Route Networking and minimum routes are found using Kruskals and Prims Algorithm.
U03, U04, U05, U06 ,U13,U18	Queue	Orders are entities which require a FIFO order. An order which is placed first needs to be serviced first.
U07,U08,U10,U26	Link List	To Store Product, multiple keys can be same. So this rules out BST usage. As far as Queues/Stacks are concerned, they require a FIFO/LIFO principle respectively. There is no such principle required. So, Link List is used.
U14	Link List	To Store Users. Justification reason the same as above.
U12,U15	Graphs	A graph us helpful in route plotting by edge and vertex connectivity.
U17	Binary Search Tree	To Store Rider Fuel Bills, A BST is used. The biggest limitation of BST is unique values. To abate this problem, a provision is inserted so that each time a fuel log is entered, a unique value is entered.

1.12 Exceptions

An exception in computer programming is a special condition encountered during program execution that is unexpected or anomalous. An exception occurs when a program attempts to open a file that does not exist or encounters a read error. This may also occur if any unexpected behaviour is encountered on runtime.

To avoid a fatal error, the programmer must anticipate exceptions and properly handle them in the program code, branching program execution as needed. Exception handling is a feature of computer programming.

Table 1.49: Possible Exceptions in system

Type of Exception	Why this exception will occur	Use Case Id in which exception could be occurred	How you will handle the exception
Runtime	Incomplete/Empty Data	All	Reload operation
No such Field	Product is out of stock / incomplete entries	U7,U8,U10,U26	Reimbursement, logging, notices
Runtime	Two orders of same composition are made at the exact same time	U4	Rearranging the orders
Interrupted	Maps/ Jazzcash API doesn't work correctly	U9,U15	Reload operation
Runtime	Discrepancy in rider fuel logs	U17	Cancel, log to admin
Format Error	Wrong Data, doesn't fit according to validators	All where validation is required	Inform the end user about the mistake
Location	Shop location is either occupied or doesn't exist	U2, U24	For maximum security layer, shop will not be allowed to add in the system
Chart Plotting Error	Chart is asked to plot between invalid pair of values	U28	Chart plotting operation is closed on account of incompatibility

1.13 Project Plan

1.14 Data Storage

File handling will be used by us as a mechanism for data storage in this project. The format for this storage will vary as per requirements, we will use both csv files and txt files.

1.14.1 Parser.txt

Table 1.50: parser.txt Text File

File Name	parser.txt
File Type	Text File
Data Format and values	Variable data entries, contains two lists of any data type in two lines in comma separated forms
Special Purpose *(if any)*	Transfer data from front end to grapher.py Python file to display charts *(see Analytical Reports section)*

1.14.2 CurrentOrders.csv

Table 1.51: CurrentOrders.csv

File Name	CurrentOrders.csv
File Type	csv file
Data Format and values	Column A : List<Line Item> Column B: Shop Name Column C: Shop Address Column D: Shopkeeper Name Column E: Shopkeeper Contact Column F: Rider Name Column G: Rider Contact Column H: Bill
Special Purpose (if any)	None

1.14.3 Products.csv

Table 1.52: Products.csv

File Name	Products.csv
File Type	csv file
Data Format and values	Column A: Name Column B: Price Column C: Tax Ratio Column D: ID Column E: Category Column F: Is Perishable
Special Purpose (if any)	None

1.14.4 Users.csv

Table 1.53: Users.csv

File Name	Users.csv
File Type	csv file
Data Format and values	Column A: Username Column B: Password Column C: Role Column D: CNIC Column E: Address Column F: Phone Number
Special Purpose (if any)	None

1.14.5 Shops.csv

Table 1.54: Shops.csv

File Name	Shops.csv
File Type	csv file
Data Format and values	Column A: Shop Name Column B: Shop Address Column C: Shop City Column D: Shop Area Column E: Shop Contact Column F: Shopkeeper Name Column G: Shopkeeper Email Column H: Latitudes Column I: Longitudes
Special Purpose (if any)	None

1.15 Email Sending

1. An email will be sent at order placement. It will be a short email having content, " Greetings customer **XYZ**, an order has been placed under order id *ABC11*. Thanks for using our distribution network".

2. An email will be sent at order payment. It will be a short email having content, " Greetings customer **XYZ**, an order id **ABC** has been paid under payment id **XYZ** using *method of payment*. Thanks for using our distribution network".

1.16 Analytical Reports

In Analytical reports, we will generate 5 types of reports.

- An order report for the admin dashboard to verify sales.The format of this report will be a pdf file, availabale for download. It will contain a traffic bar chart.
- Profit Loss Trends. This will be a pdf file, available for downloads. This will be a graph chart.
- Sales Number Trends. Format and description same as above.
- Rider Trends.This will be an executive summary of all orders worth against each rider, having a bar chart format.
- Rider Fuel Logs. This will be a pie chart, not available for download, It will contain a weekly report of all rider fuel expenses.