Daily Need Distribution System

CS262- Design Document

Project Supervisor

Mr. Samyan Qayyum Wahla

Project Members(G28)

Name: M. Bilal Baig Registration No. 2021-CS-36

Name: Hassan Wasti Registration No. 2019-CS-40

Name: Afaq Ahmad Registration No. 2019-CS-248

Department of Computer Science

University of Engineering and Technology, Lahore

Pakistan

Table of Contents

Project Description:

The project is a facility for a better supply and receive of the products. The project is about a company or any supply source which wants the best supply and communication between the client and the source, also they want to make their product deliveries more fluent and quicker so that they may get a better response from the client. In this regard they communicate with their deliverer, have track of his location and provide him the facility of the shortest route to the client. They have the record of the deliveries that are done by the deliverer or pending by and have a better communication between them and their deliverer. Once the order is delivered the system marks that and reminds the deliverer for the pending ones if any. Also, the deliverer has the feature to communicate with the client itself and take order and submit the order to the company/supplier, which makes the client more confident and he feels like the company is more responsible and serious about the orders. Along with taking orders the system also manages the payment schedule from the client and has a complete record of the payment pending and paid. The payments can pay manually or online. These payment methods also provide a facility to the client to pay of his own choice. All these features lead to a better flow of supplying the products to the client and make a good use of fuel makes profit to the company/supplier.

The employee will have his own window the end users will be the rider and the shopkeeper. They will also have their own respective Window. On the employee's window he will be able to manage the rider and the shopkeeper, the messages, products (including shopkeeper orders list and stock in the company) and the payments from the shopkeepers. The rider will be able to see the pending orders as well as the delivered ones. He will also be able to see the roots get the shortest path to visit all the shops to minimize the cost of the fuel. This feature will be provided by the company. The communication with the employee and taking order from the shopkeeper will also be managed. In the shopkeeper window he will manage the communications with the company as well as the employee. He will be able to watch the orders he gave and will rate the company and the rider.

Project Features:

- Our **Daily Need Distributed System** (**DNDS**) will provide a short and proper path to the Rider.
- Our (**DNDS**) will provide a complete information of the Rider to the Shopkeeper. So, it prevents any scam.
- Our (**DNDS**) will help company to check whether the delivery is done or pending.
- Our (**DNDS**) will help to keep eye on the rider that it supplies the stock or not.
- Our (**DNDS**) will help Shopkeeper to add reviews on rider behavior.
- Our (**DNDS**) will help Shopkeeper to send message/email.
- Our (**DNDS**) will help Shopkeeper to place order. When he goes to the shortage of the stock in his shop.
- Our (**DNDS**) will provide complete Product list. So, it will help them to take order from the shopkeeper.

Technology Stack:

Language	
Windows Form	We use windows form for the frontend of our project.
C#	We use C# language for the backend of our project.

Project Actors:

Project actors which directly or indirectly uses the system are the followings:

1. **Admin:**
   Admin will add/edit/delete rider or shopkeeper. It will set the conditions for the other actors.

2. **Rider:**
   Rider will see the details about the whole day delivery.

3. **Shopkeeper:**
   Shopkeeper will see the arrival of the rider and send messages to the rider and company about the services.

Stakeholder:

Stakeholders are all those parties which are directly or indirectly linked to a company. Company policies affect them and company should keep all of them in mind while making policies.

Use Cases:

1. Admin:

- Admin can add/delt/edit rider and shopkeeper.
- Admin can display ad to the website.
- Admin will help user if they get stuck at anywhere.
- Admin can Login/Register to the system.
- Admin can update profile.
- Admin can change password.

2. Rider:

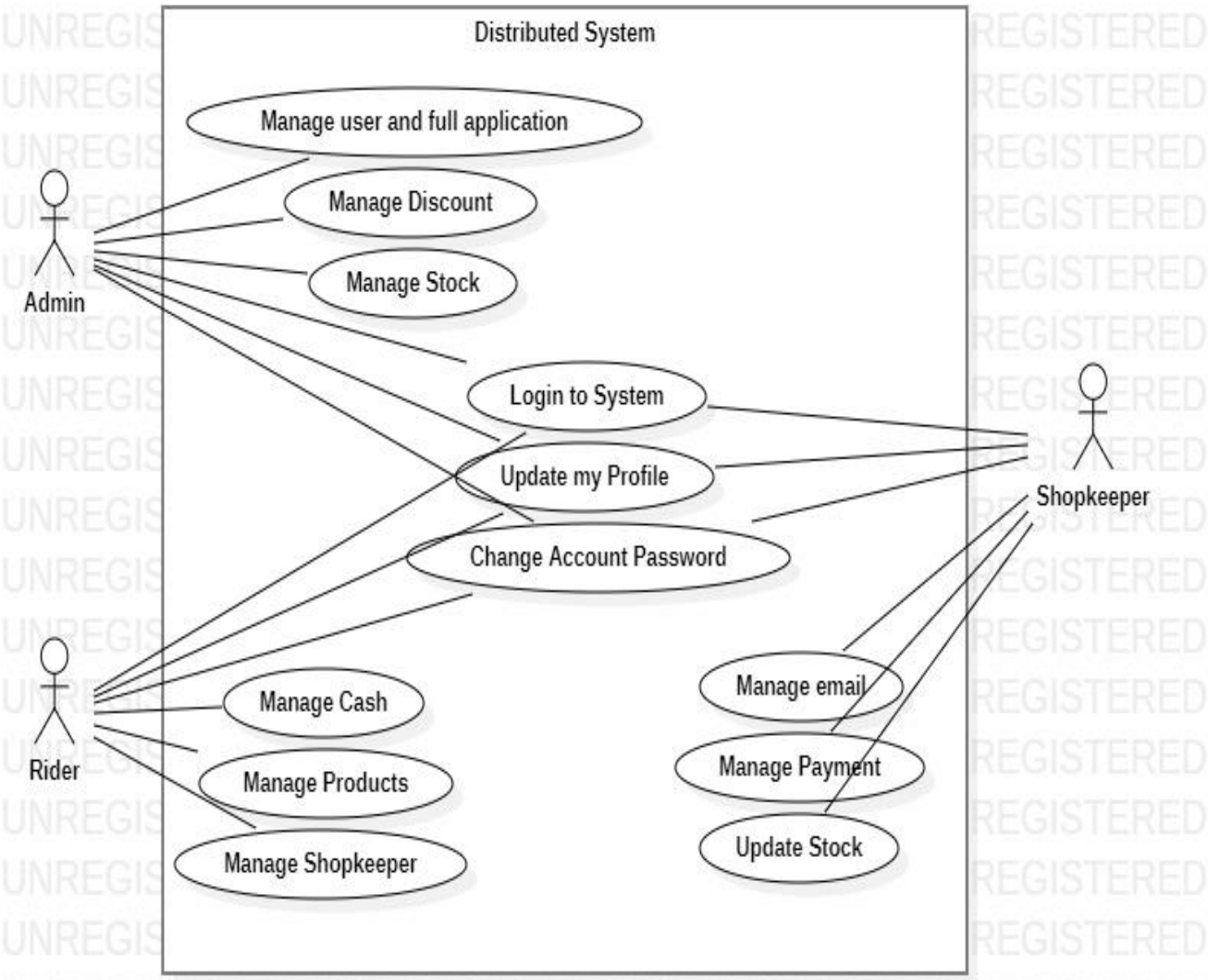
- Rider can Login/Register to the system.
- Rider can update profile.
- Rider can change password.
- Rider can manage shopkeeper.
- Rider can manage cash.
- Rider can manage stock.

3. Shopkeeper:

- Shopkeeper can Login/Register to the system.
- Shopkeeper can update profile.
- Shopkeeper can change password.
- Shopkeeper can send email/message.

- Shopkeeper can check stocks.

Use Case Diagram:



Use Case:

Use Case ID	**U01**
Name	**Splash Screen**
Actor	Rider, Shopkeeper, Rider.
Description	Splash Screen will open when the actors of our software when ever open the DNDS will see the first of all the splash screen. It has the process bar which complete to 100% then it goes to the next screen and process.
Flow	1. Actor First of all install and download the desktop application to his gadget.

	2. Then he will open the Desktop app. 3. First, he will see this screen in front of him showing the company name and welcoming him.
Use Case ID	**U02**
Name	**Registration for rider, Shopkeeper, Admin.**
Actor	Rider, Shopkeeper, Rider.
Description	Registration Process is for every actor to track every one reaction and record. After completing the process of splash screen. It will take them to register screen to register themselves.
Flow	1. Actor will enter their confidential according to the requirement to complete the process. 2. After completing the process, it will take them to the next screen.

Use Case ID	**U03**
Name	**Login for rider, Shopkeeper, Admin.**
Actor	Rider, Shopkeeper, Rider.
Description	When ever Actor open the desktop application. He will enter the confidential which he gave during the registration process. It will keep his account safe from different activities.
Flow	1. Actor will open the desktop application. 2. Then he enters his confidential to the Login screen. 3. After enter the right information. 4. It will take him to the next screen.

Use Case ID	**U04**
Name	**Admin Panel.**

Actor	Admin.
Description	Admin will Choose the given option from the required ones. Then click on it. It will take him to the next screen. Like if he clicks on the Shopkeeper, it will take him to the shop keeper panel where he can choose the option for shopkeeper.
Flow	1. Admin complete the process of login. 2. He will see the interface. 3. Where he can choose the one from Rider/Shopkeeper. 4. After choosing it. He will go to the next screen.

Use Case ID	**U05**
Name	A**dmin panel for Shopkeeper.**
Actor	Admin.
Description	After selecting Shopkeeper. Admin will see the new interface where he can enter the next required option. Then select the Enter option. It will take him to the new Shopkeeper interface.
Flow	1. On selecting the option of Shopkeeper from the **Admin Panel.** 2. Now Admin will select the required option from Shopkeeper admin option. 3. Then he performs the add/ edit/delt and other option from this panel.

Use Case ID	**U06**
Name	**Add Shopkeeper by Admin.**
Actor	Admin
Description	Whenever new shopkeeper is added in the DNDS. Then Admin will add the shopkeeper by taking the information from it. It will help him to place order and do activities in the desktop application.

Flow	1. After clicking on the Add Shopkeeper. 2. Admin will see the interface where all fields will display to them. 3. Admin Add all information about the shopkeeper. 4. After adding information click on the Enter button. 5. All information is added to the database. 6. Now shopkeeper do message and perform activities.

Use Case ID	**U07**
Name	**Delete Shopkeeper by Admin.**
Actor	Admin
Description	Whenever there is the need to delete the shopkeeper in the DNDS. Then Admin will delete the shopkeeper. This use case will help him to delete the shopkeeper.
Flow	1. After clicking on the Delete Shopkeeper. 2. Admin will see the interface where all fields will display to them. 3. Admin Delete the shopkeeper. 4. Shopkeeper is Deleted to the database.

Use Case ID	**U08**
Name	**Edit Shopkeeper by Admin.**
Actor	Admin
Description	Whenever there is need to edit any information about the shopkeeper. Admin will edit the information. Like, if shopkeeper changes his number or email. His confidential will be changed by the admin.
Flow	1. After clicking on the Edit Shopkeeper. 2. Admin will see the interface where all fields will display to them. 3. Admin Edit information about the shopkeeper. 4. After editing information click on the Enter button.

	5. All information is edited to the database.

Use Case ID	**U09**
Name	**Add Rider by Admin.**
Actor	Admin
Description	Whenever new rider is added in the DNDS. Then Admin will add the rider by taking the information from it. It will help him to deliver order to the shopkeeper and do activities in the desktop application.
Flow	1. After clicking on the Add Rider. 2. Admin will see the interface where all fields will display to them. 3. Admin Add all information about the Rider. 4. After adding information click on the Enter button. 5. All information is added to the database. 6. Now rider place order to the shopkeeper and perform other activities.

Use Case ID	**U10**
Name	**Edit Rider by Admin.**
Actor	Admin
Description	Whenever there is need to edit any information about the rider. Admin will edit the information. Like, if rider changes his number or email. His confidential will be changed by the admin. He sends any message to the admin then admin will change things.
Flow	1. After clicking on the Edit Rider. 2. Admin will see the interface where all fields will display to them. 3. Admin Edit information about the rider. 4. After editing information click on the Enter button. 5. All information is edited to the database.

Use Case ID	**U11**
Name	**Delete Rider by Admin.**
Actor	Admin
Description	Whenever there is the need to delete the rider in the DNDS. Then Admin will delete the rider. This use case will help him to delete the rider.
Flow	1. After clicking on the Delete Rider. 2. Admin will see the interface where all fields will display to them. 3. Admin Delete the Rider. 4. Rider is Deleted to the database.

Use Case ID	**U12**
Name	**Shopkeeper Order Details.**
Actor	Admin
Description	Purpose of this use case is to view the orders of the shopkeeper by the admin. This use case will help to view which order and product will need to submit by the rider. So, it is made available by the staff. After that it will deliver to the shopkeeper on time.
Flow	1. Admin will open the order details of the Shopkeeper. 2. Admin send order to the Wearhouse team to make it available. 3. So, it is sent to the shopkeeper.

Use Case ID	**U13**
Name	**Assign order to the rider for delivery.**
Actor	Admin
Description	This use case will help the admin to assign the rider to the order. When one rider

	punches the order and place it to the DNDS. It will give to the next delivery rider to deliver it. It will help the rider to send the order accordingly.
Flow	1. Admin go to the Shopkeeper dashboard. 2. Select the Assign order to the rider for delivery. 3. Then new interface will open. 4. Where Admin will assign rider to waiting order.

Use Case ID	**U14**
Name	**Product List.**
Actor	Admin
Description	This use case is made for the rider. It will help the rider to take the order from the shopkeeper easily. Product list has almost all of the items which company is selling. It also has the updated price.
Flow	1. Admin will open the product list item from the rider dashboard. 2. Admin see the products and prices. 3. Admin will change the product and prices accordingly.

Use Case ID	**U15**
Name	**Check Reviews.**
Actor	Admin
Description	This use case is helpful for the shopkeeper to give the reviews about the rider. This also keep the rider behavior under control. Admin will able to see the reviews and make decision accordingly.
Flow	1. Admin will go to the shopkeeper dashboard. 2. Select the Check Reviews option from the dashboard. 3. Then interface will open to the messages.

	4. Where Admin see the messages to the rider and shopkeeper.

Use Case ID	**U16**
Name	**Location of Shop.**
Actor	Rider
Description	Rider needs the location and other details of the shop. So, they can deliver the delivery. Rider will get the shortage path. Rider also sees the other details on the screen. It will help him to find the shop. After finding the shop then all details like price and other things will display to him.
Flow	1. Rider first goes to the Dashboard. 2. Then click on the Location of the shop. 3. He will see the interface of location in front of him.

Use Case ID	**U17**
Name	**Order Taking.**
Actor	Rider
Description	The rider will reach to the assign shopkeeper. He will bring his device out and start taking order from the shopkeeper that he will deliver that order in the future.
Flow	1. Rider first goes to the Dashboard. 2. Then click on the Order Taking button on the Dashboard. 3. Interface will open. 4. Rider takes the order from the shopkeeper. 5. On pressing the Done button. It will save to the Database.

Use Case ID	**U18**
Name	**Messages from Rider and Shopkeeper.**
Actor	Rider and Shopkeeper

Description	This use case is important because admin will see messages from rider and shopkeeper. He will take actions accordingly. So, these two actors will send their good or bad reviews then admin we behave accordingly.
Flow	1. Rider and Shopkeeper go to the Dashboard. 2. Press the Message Button. 3. New interface will open in front of him. 4. He writes message and send it to the admin.

Use Case ID	**U19**
Name	**Complete and incomplete order.**
Actor	Rider
Description	Rider will see his all orders. Orders which he completed or which he is not completed. This will help him to manage his order in a line. Like if he completed 2 orders, he will see that and he has two orders remaining to complete he will see that also.
Flow	1. Rider will go to the dashboard. 2. Click on the details of the order. 3. Where he sees the Detail about the order. 4. Completed or incomplete orders detail.

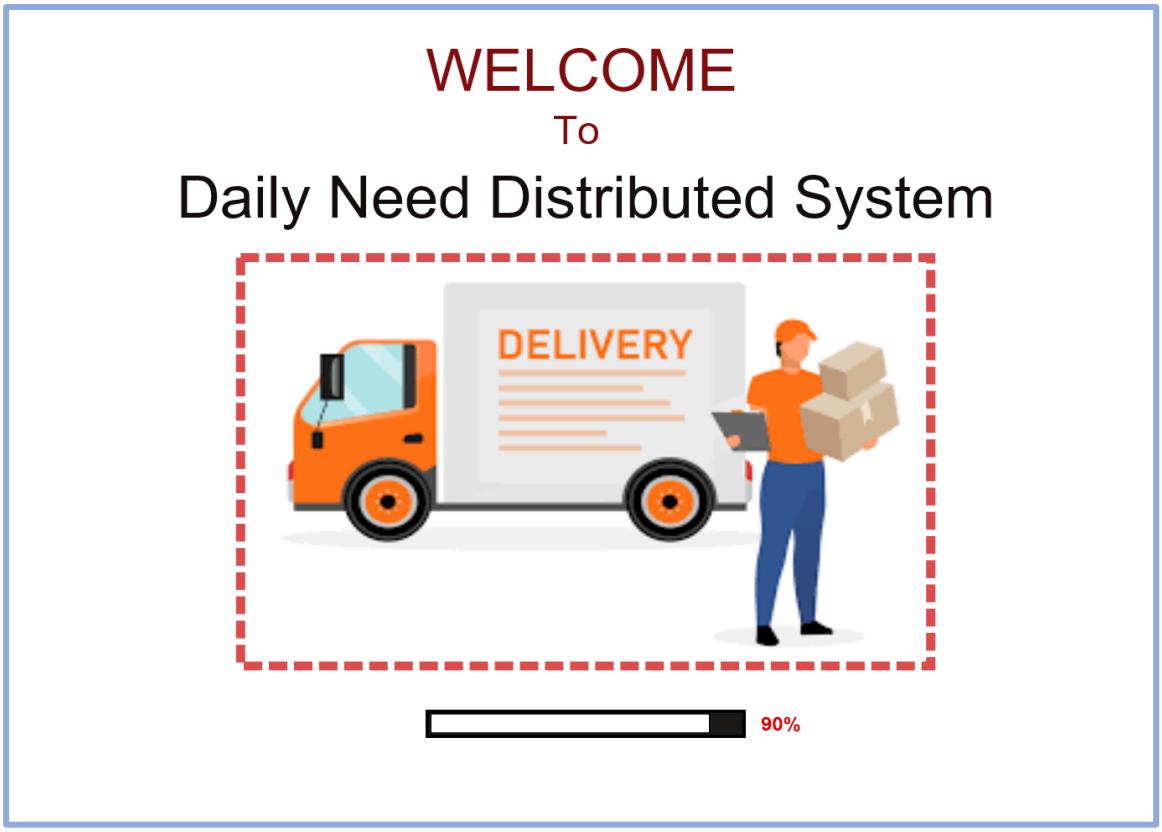
Use Case ID	**U20**
Name	**Shopkeeper Reviews**
Actor	Rider
Description	This use case will help Shopkeeper to give reviews on the rider and the product of the rider delivered. This helps the admin to track the behavior of the rider with the shopkeeper and the product delivered to the shopkeeper is in the right form.
Flow	1. Shopkeeper will go to the Dashboard. 2. Where he selects the reviews option.

	3. Add the reviews and press the enter button. 4. Reviews will send to the admin.

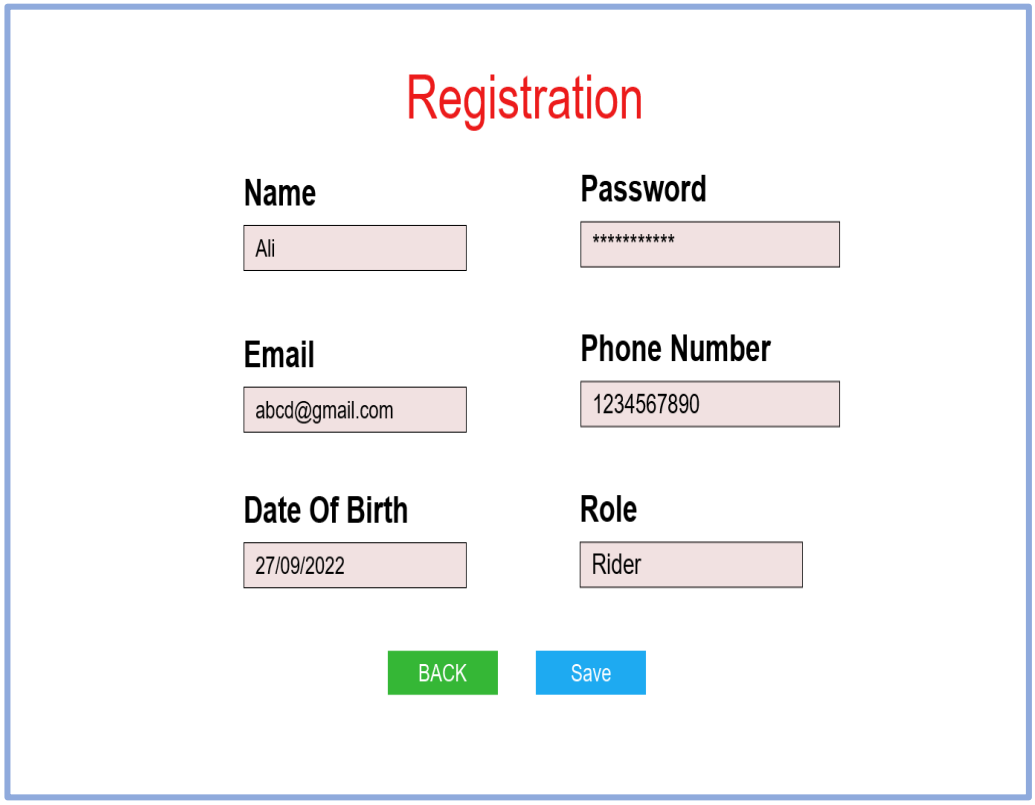
Use Case ID	**U21**
Name	**Shopkeeper Stock**
Actor	Rider
Description	This use case will help the shopkeeper to place the order. When he faces the shortage of the items in his shop. This message will help the admin to send the rider to take order from the shopkeeper. Then ready the items and place the delivery back to the shopkeeper
Flow	1. Shopkeeper will go the Dashboard. 2. Select the add stock option. 3. Place the message to the Admin. 4. Admin will see his message then send the delivery guy.

Use Interfaces:

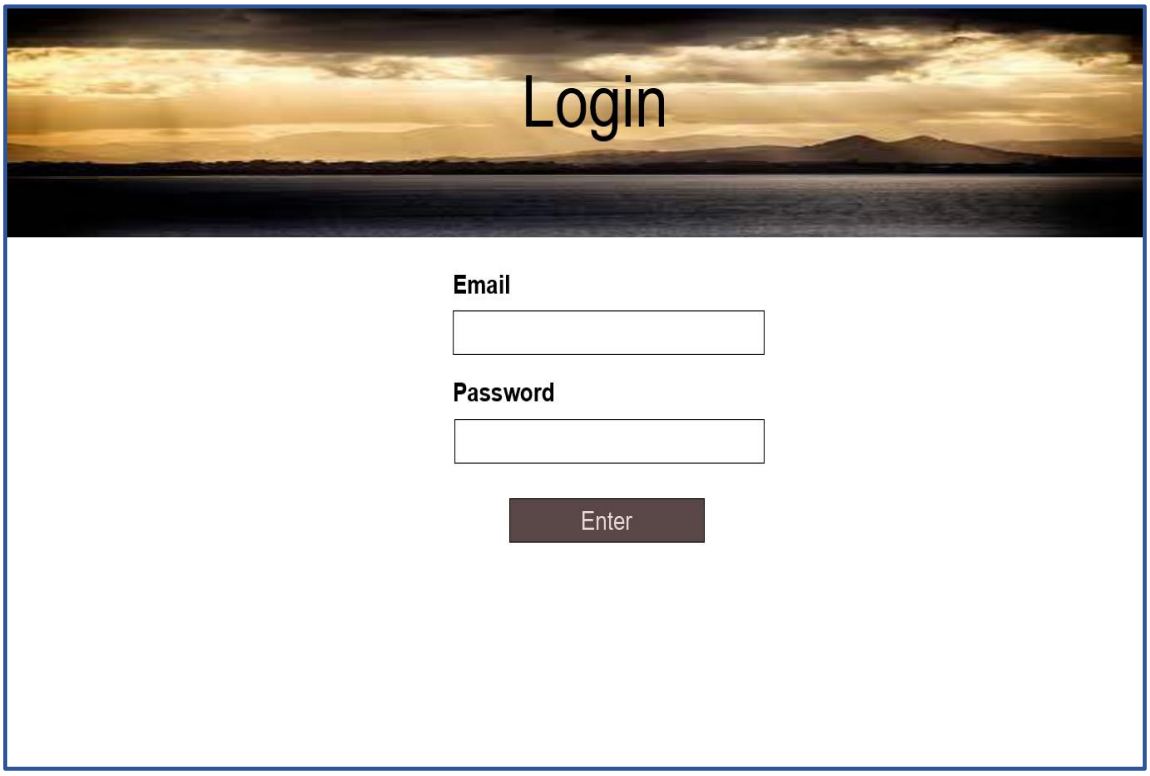
Interface ID	**I01**
Name	Splash Interface.
Linked Use Case	U01

UI Screen	
Validators	No Validation Required

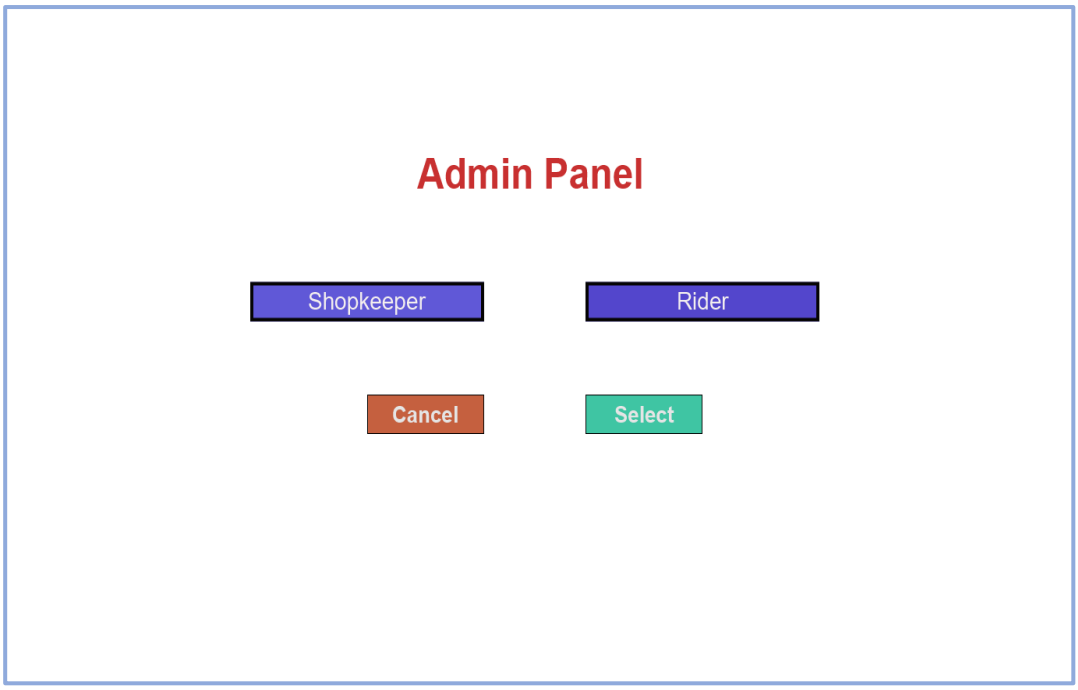
Interface ID	**I02**
Name	**Registration for rider, Shopkeeper, Admin interface.**
Linked Use Case	U02

UI Screen	
Validators	All of the input fields should be filled with the relevant text. Like we talk about email it is filled with the proper email.

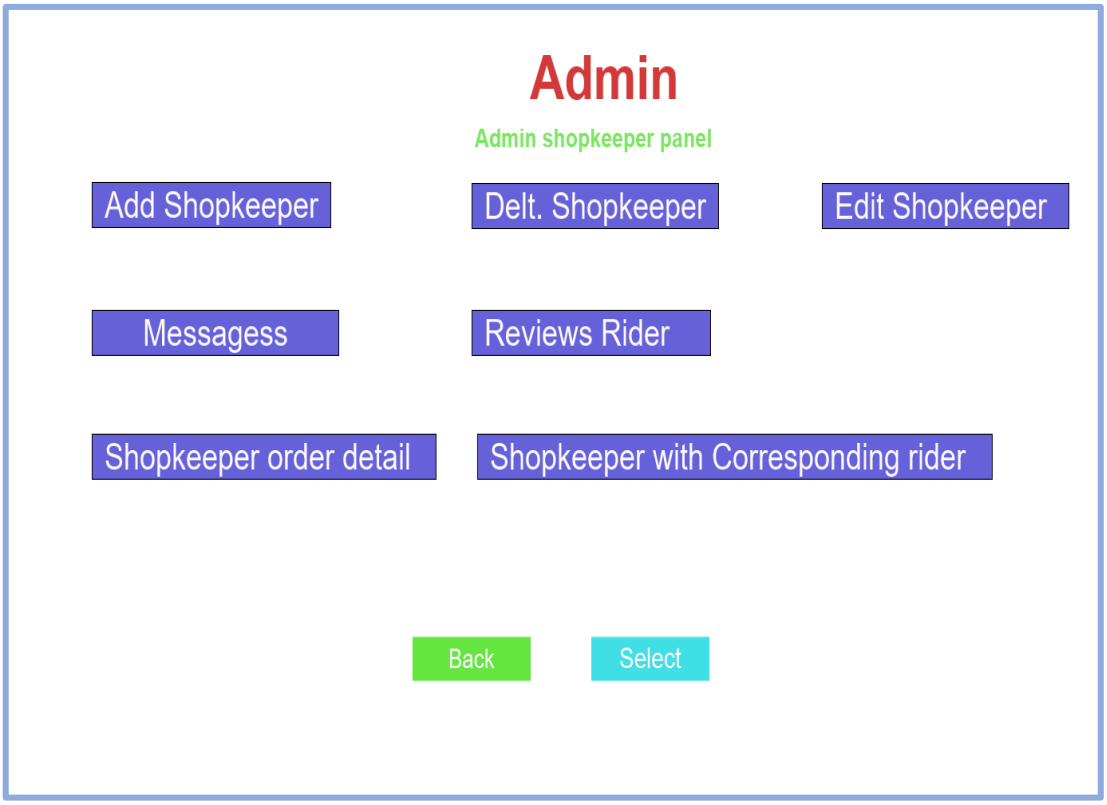
Interface ID	**I03**
Name	**Login for Shopkeeper, Rider, Admin interface.**
Linked Use Case	U03

UI Screen	
Validators	Actors will enter email and password of themselves after putting it here, they will go to the respective page after completing the process.

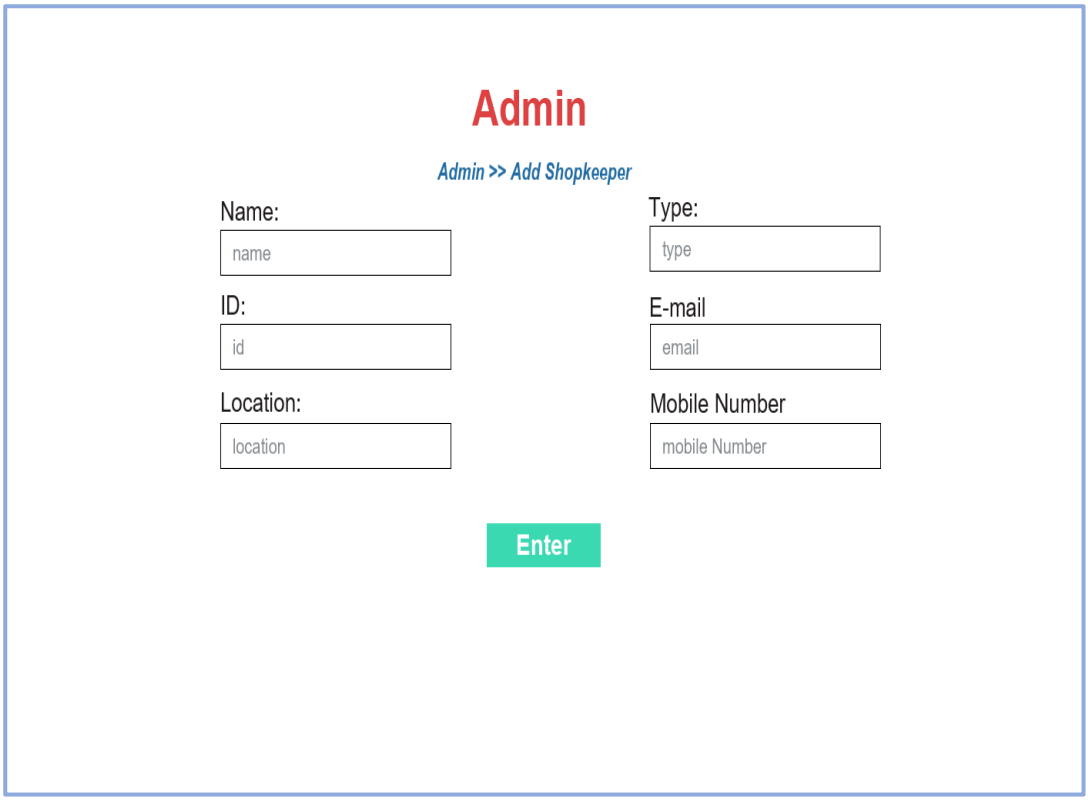
Interface ID	**I04**
Name	**Admin Panel Interface.**
Linked Use Case	U04
UI Screen	

	
Validators	Admin select one option then enter the select button. It will take him to the next screen.

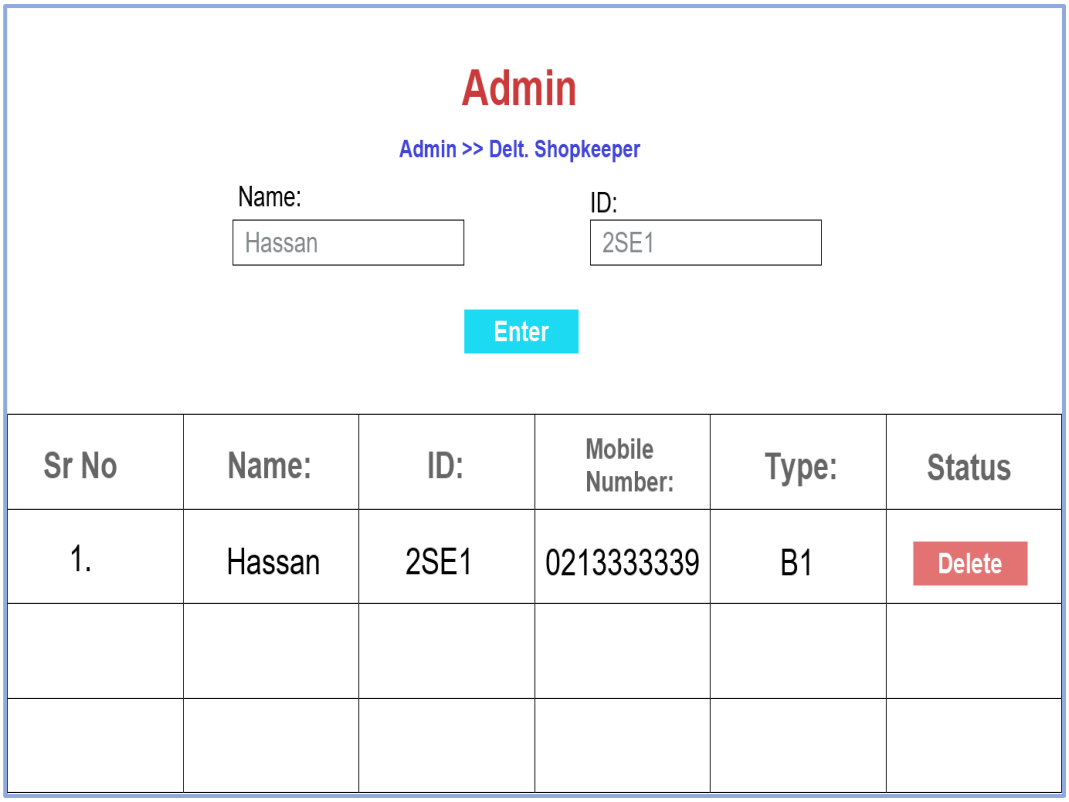
Interface ID	**I05**
Name	**Admin panel for Shopkeeper.**
Linked Use Case	U05

UI Screen	
Validators	Admin click on the required option then proceed to do further Process.

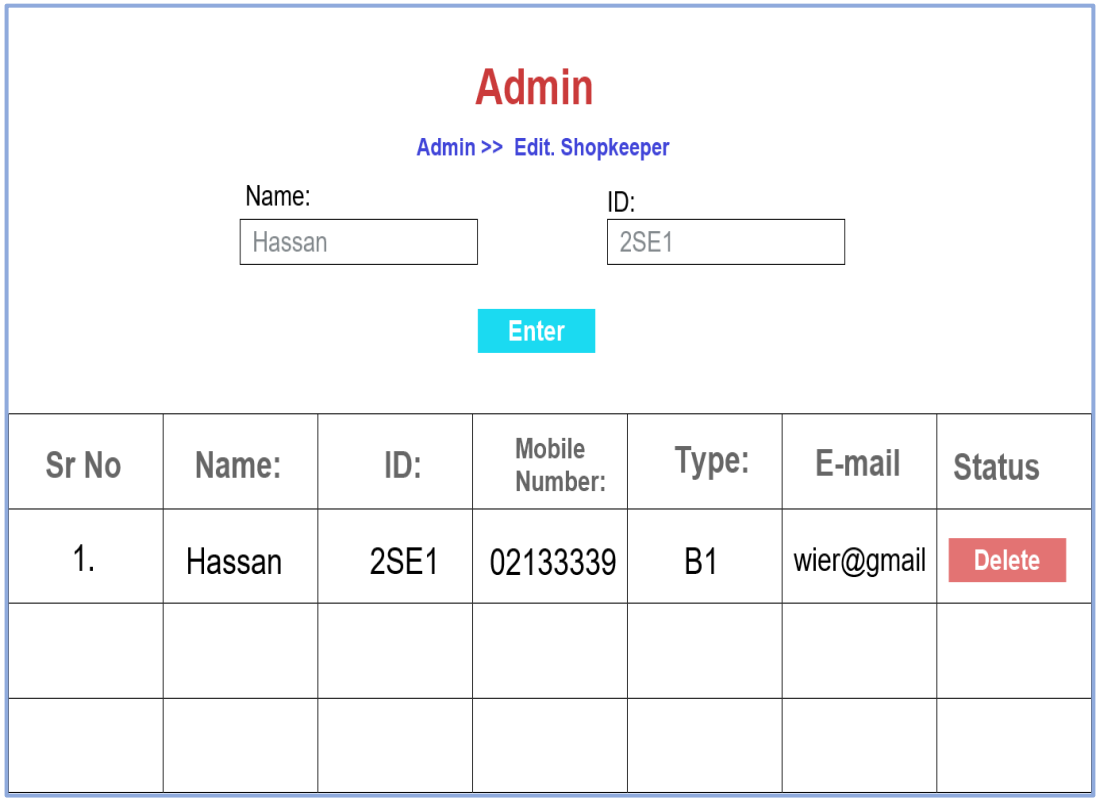
Interface ID	**I06**
Name	**Add Shopkeeper by Admin Interface.**
Linked Use Case	U06

UI Screen	
Validators	Admin add all information of the Shopkeeper. Then press enter to save the option to the database.

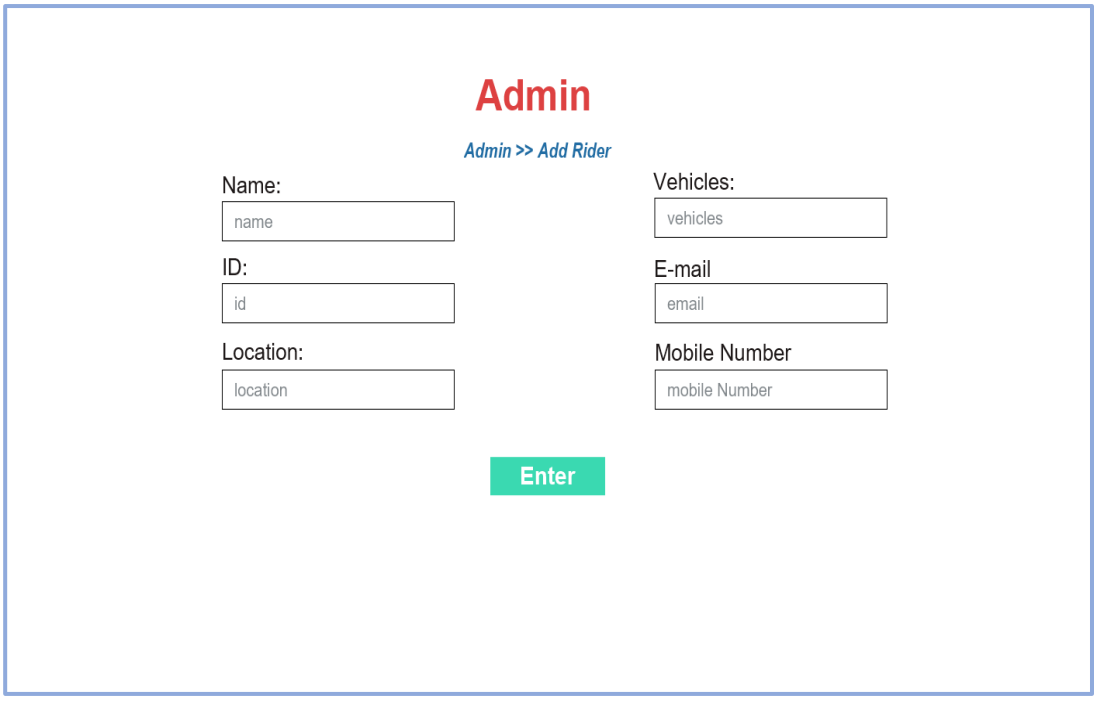
Interface ID	**I07**
Name	Delete Shopkeeper by Admin
Linked Use Case	U07

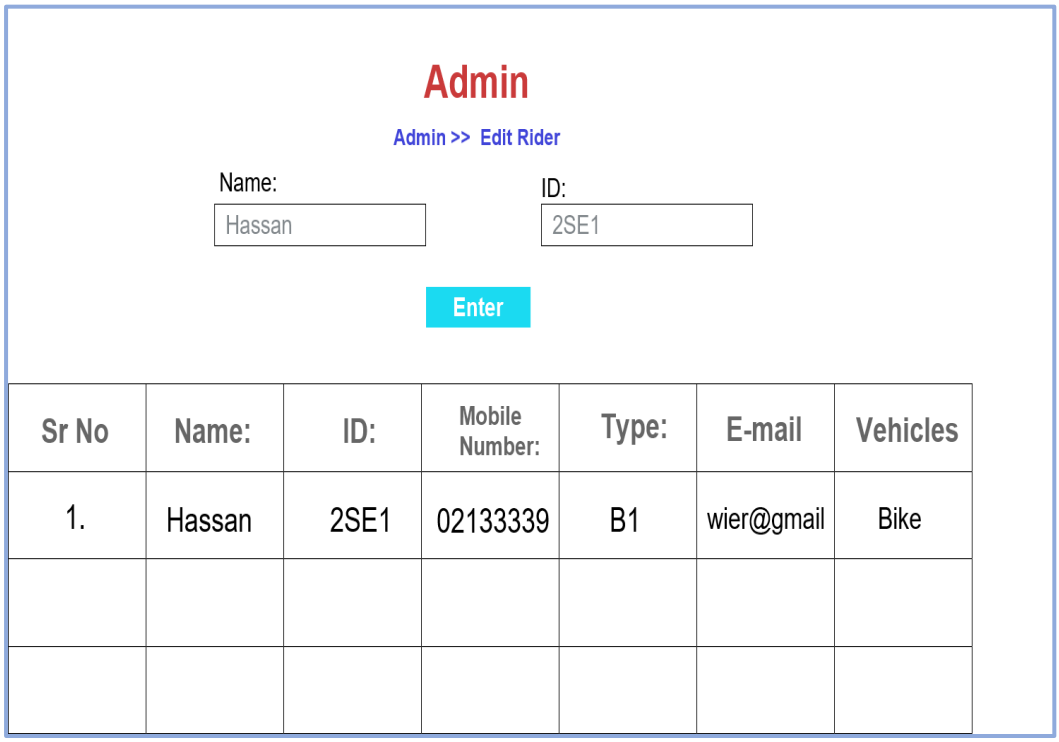
UI Screen (JustInMind)	
Validators	Admin will enter the name and id of the shopkeeper then press enter button then all information display where admin can delete the person. If he changed his mind.

Interface ID	**I08**
Name	**Edit Shopkeeper by Admin**
Linked Use Case	U08

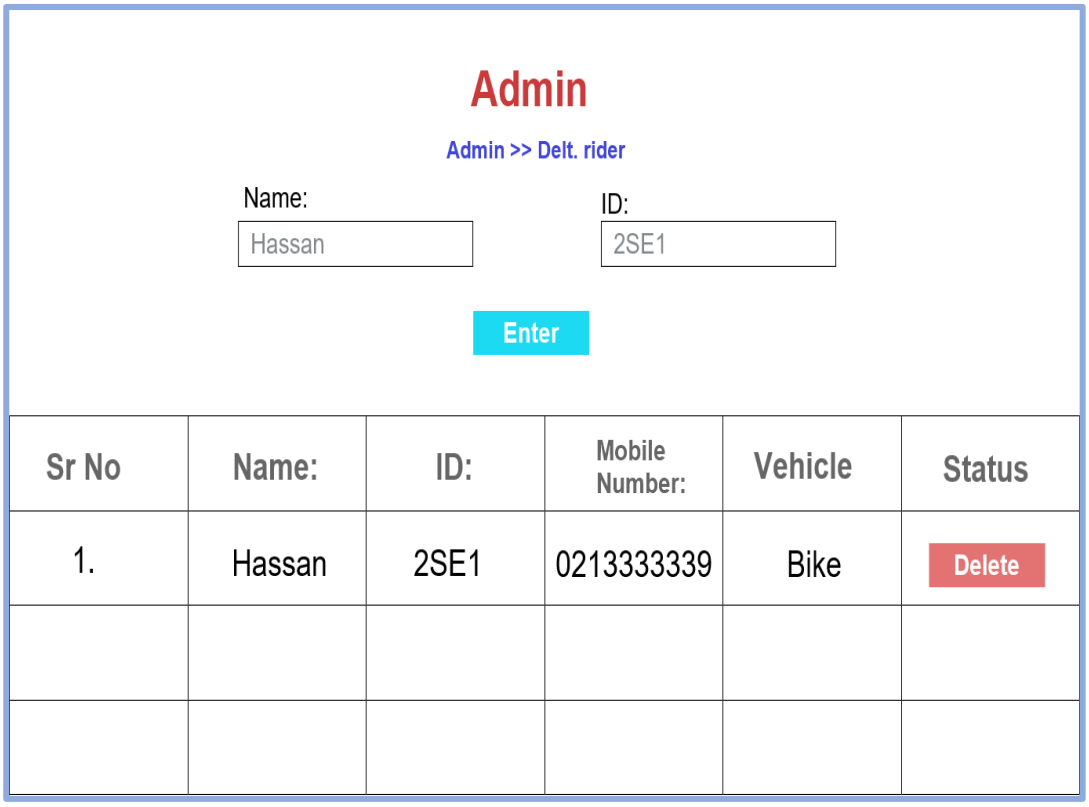
UI Screen (JustInMind)	
Validators	Admin will enter the name and id of the shopkeeper then press enter button then all information display. Where admin can edit the person.

Interface ID	**I09**
Name	**Add Rider by Admin.**
Linked Use Case	U09

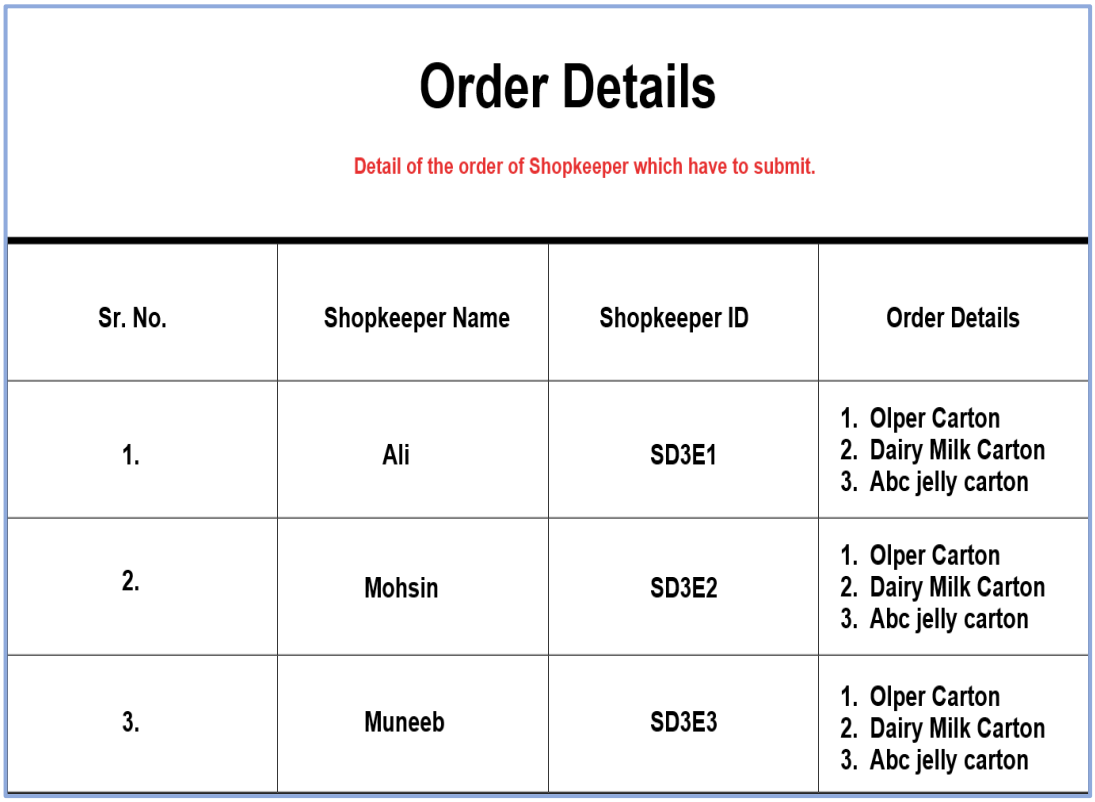
UI Screen	
Validators	Admin add all information of the Rider. Then press enter to save the record into the database.

Interface ID	**I10**
Name	**Edit Rider by Admin.**
Linked Use Case	 U10
UI Screen (JustInMind)	

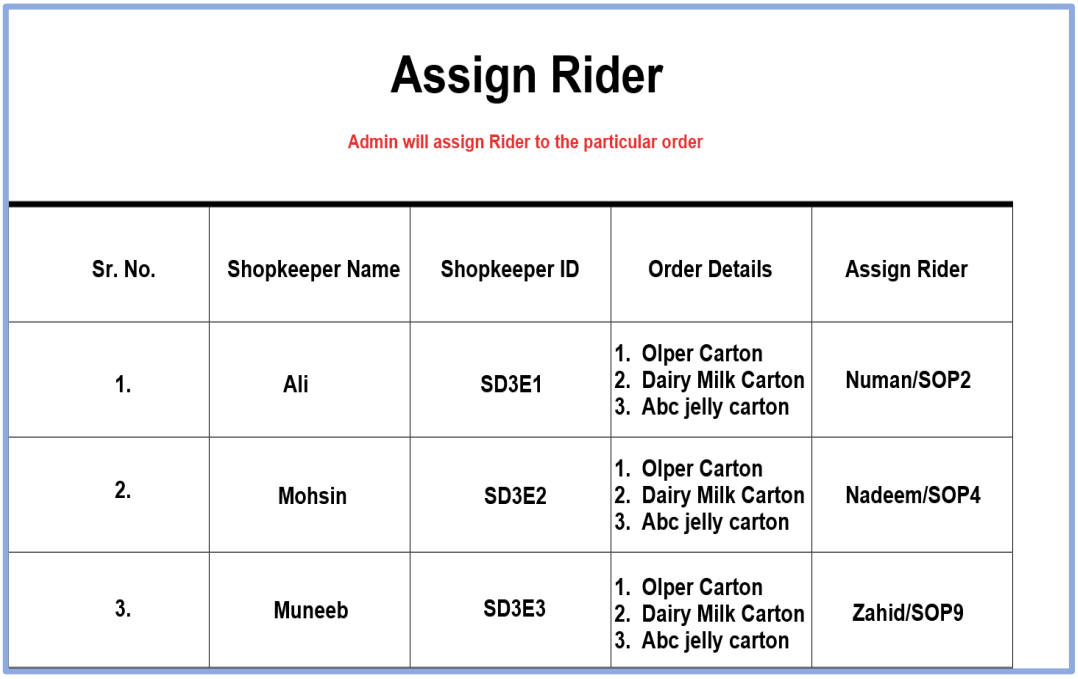
Validators	Admin will enter the name and id of the rider then press enter button then all information display. Where admin can edit the person/rider.

Interface ID	**I11**
Name	**Delete Rider by Admin**
Linked Use Case	U11
UI Screen (JustInMind)	
Validators	Admin will enter the correct name and id of the rider then press enter button then all information display. Where admin can delete the person/rider.

Interface ID	**I12**
Name	**Shopkeeper Order Details**
Linked Use Case	U12

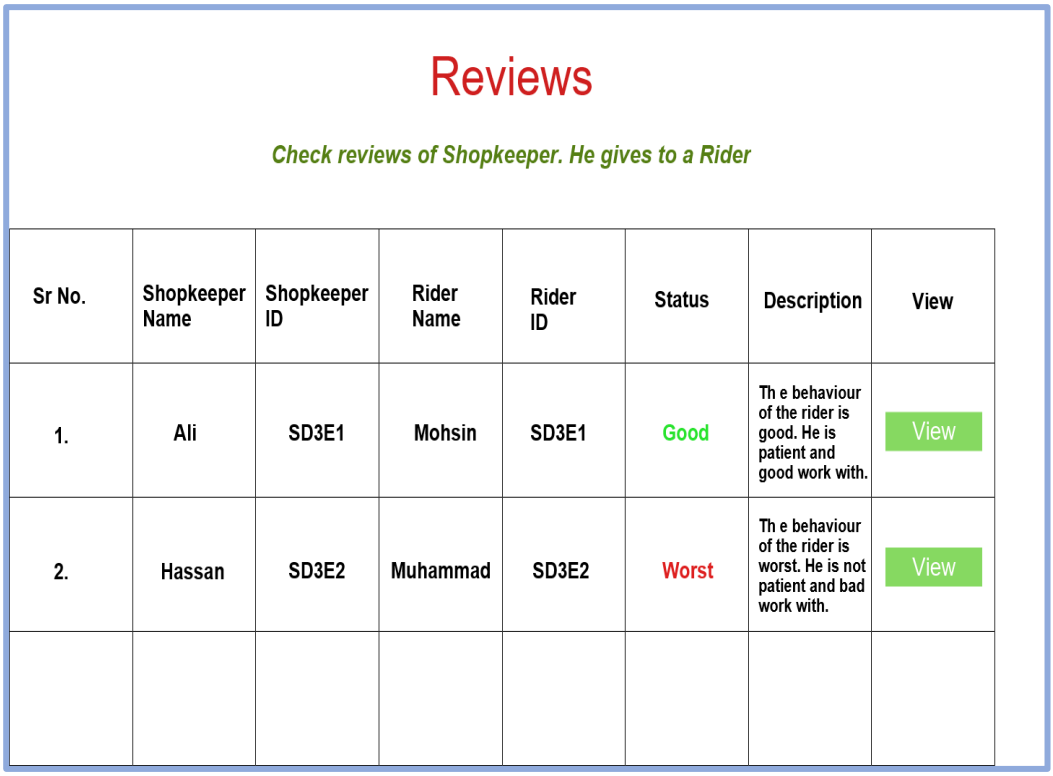
UI Screen	
Validators	No validation required because it is the information which is added by the rider. It just displays to the admin.

Interface ID	**I13**
Name	**Assign order to the rider for delivery.**
Linked Use Case	U13

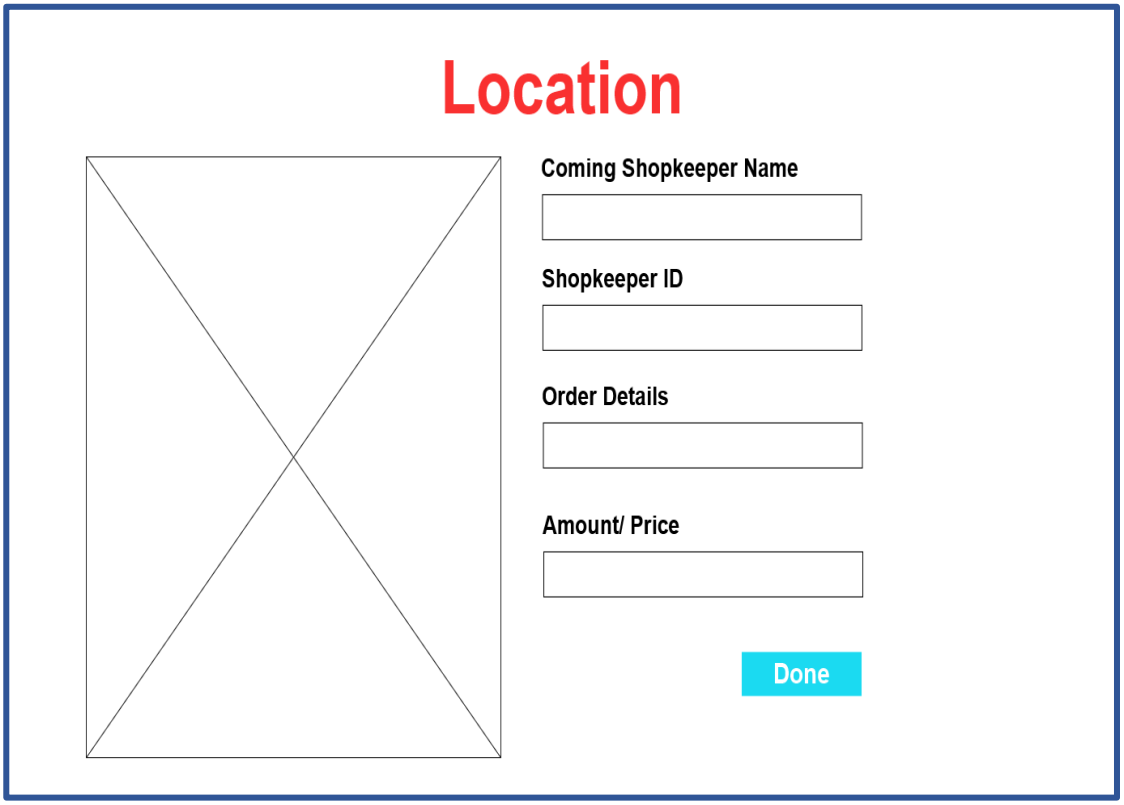
UI Screen	
Validators	Admin need to assign the nearest and available rider to the available order. Admin knows the available rider then place it here.

Interface ID	**I14**
Name	**Product List.**
Linked Use Case	U14
UI Screen	
Validators	Admin will Add/edit/Delt product accordingly. He need to add product with price and

	update that product.

Interface ID	**I15**
Name	**Check Reviews.**
Linked Use Case	U15
UI Screen	
Validators	Admin need to click on the view button to see the reviews completely and perform the action on it.

Interface ID	**I16**
Name	**Location of Shop.**
Linked Use Case	U16

UI Screen	
Validators	Rider will see the all information of the shopkeeper. It will help him to reach the shop. After reaching the shop and take money from the shopkeeper. He presses the Done key. So, the process is complete.

Interface ID	**I17**
Name	**Order Taking Screen**
Linked Use Case	U17

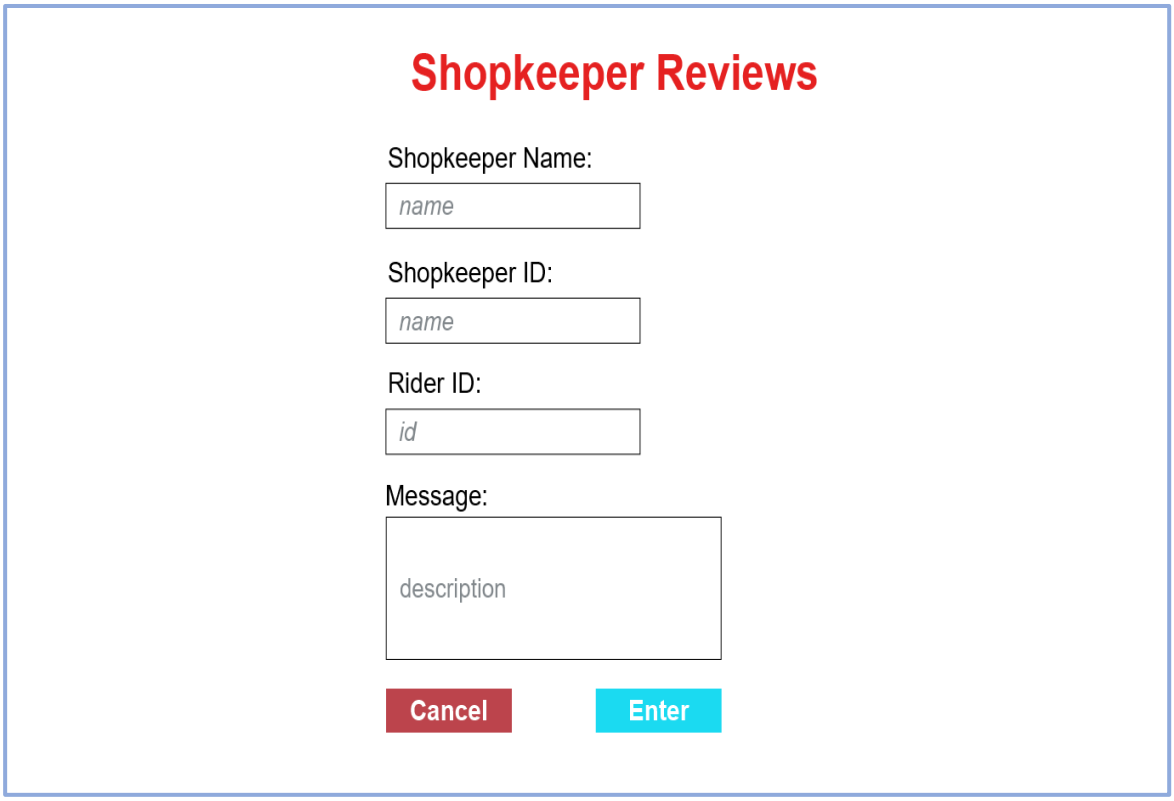
UI Screen	
Validators	Rider fills the all field to take order from the shopkeeper. After filling the all fields. It presses the enter key. Then the process of the interface is complete.

Interface ID	**I18**
Name	**Message Interface for Shopkeeper and rider.**
Linked Use Case	U18

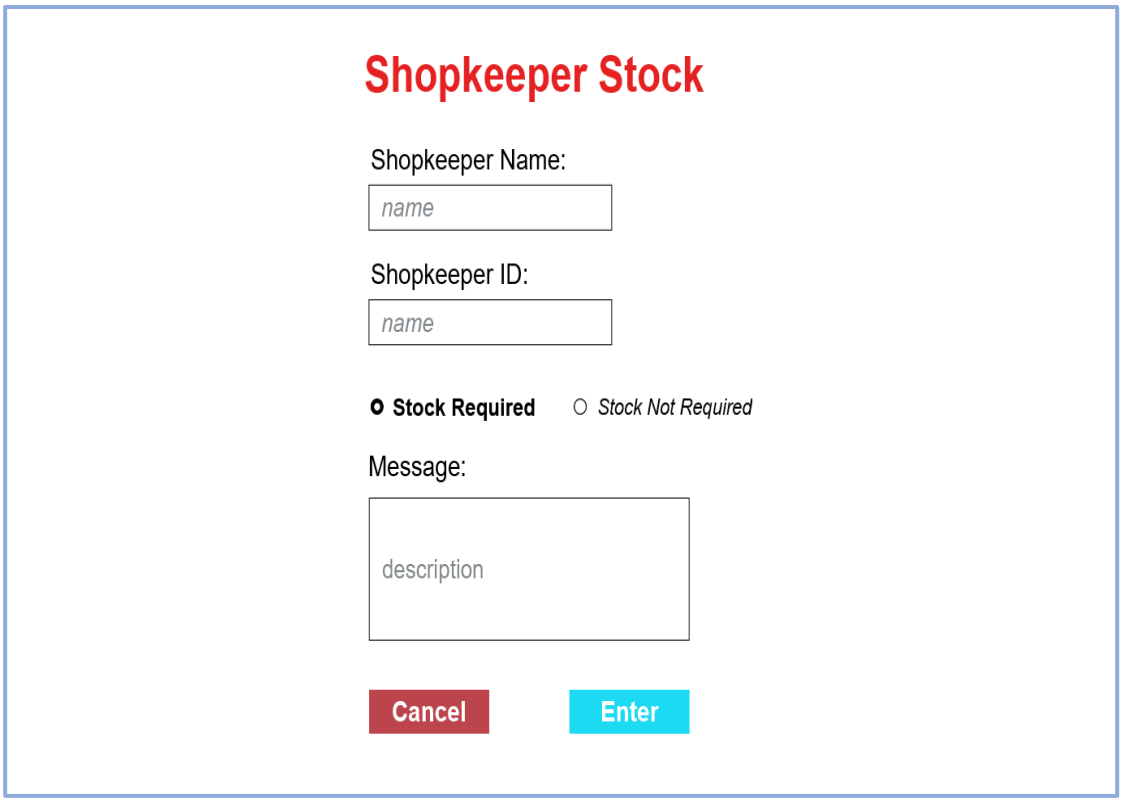
UI Screen	
Validators	No validation required

Interface ID	**I19**
Name	**Order completed or incomplete.**
Linked Use Case	U19
UI Screen	

Validators	No validation required; everything is already done.

Interface ID	I20
Name	**Shopkeeper Reviews**
Linked Use Case	U20
UI Screen (JustInMind)	
Validators	Shopkeeper add his name, id, rider id and message then place review of the rider here.

Interface ID	**I21**
Name	**Shopkeeper Stock**
Linked Use Case	U21

UI Screen (JustInMind)	
Validators	When ever shopkeeper end with the stock it will use this interface and put all information and tell company to send rider to his shop,

User Interface Details

In this section, fill the table for summary that which use case will have the required component. Inside each box, write the counts for each component. If component is not used, write zero.

Interface Id	Textbox	Dropdown	Password Box	Table	Date Field	Buttons	AutoComplete	Radio Button	Checkbox	Menu	Text Area	Progress Bar
I01	0	0	0	0	0	0	0	0	0	0	0	1
I02	1	1	1	0	0	1	0	0	0	0	1	0
I03	1	0	1	0	0	1	0	0	0	0	1	0
I04	0	0	0	0	0	1	0	0	0	0	0	0
I05	0	0	0	0	0	1	0	0	0	0	0	0
I06	1	0	1	0	0	1	0	0	0	0	1	0
I07	1	0	0	1	0	1	0	0	0	0	1	0
I08	1	0	0	1	0	1	0	0	0	0	1	0
I09	1	0	1	0	0	1	0	0	0	0	1	0
I10	1	0	0	1	0	1	0	0	0	0	1	0
I11	1	0	0	1	0	1	0	0	0	0	1	0
I12	1	0	0	1	0	0	0	0	0	0	1	0
I13	1	0	0	1	0	0	0	0	0	0	1	0
I14	0	0	0	0	0	0	0	0	0	1	0	0
I15	0	0	0	1	0	1	0	0	0	0	0	0
I16	1	0	0	0	0	1	0	0	0	0	1	0
I17	1	0	0	1	0	1	0	0	0	0	1	0
I18	0	0	0	0	0	0	0	0	0	0	0	0
I19	0	0	0	1	0	1	0	0	0	0	0	0
I20	1	0	0	0	0	1	0	0	0	0	1	0
I21	1	0	0	0	0	1	0	1	0	0	1	0

Classes:

In this section, we do not require detailed design diagram. But identify the tentative classes with the requirement Fill the following table for details. Note that class name should follow naming conventions.

Class Name	**Software/ Domain**	**Is Abstract (Yes/No)**	**Is Singleton (Yes/No)**	**Is the class will has parametrized constructor(Yes/No)**
Employee	Domain	No	Yes	Yes
Rider	Domain	No	No	Yes
Client	Domain	No	No	Yes
Payment	Domain	No	No	Yes
Products	Domain	No	Yes	Yes
Vehicle	Domain	No	No	Yes
INFO	Domain	No	No	Yes
Login	Domain	No	No	Yes
Location	Domain	No	No	Yes

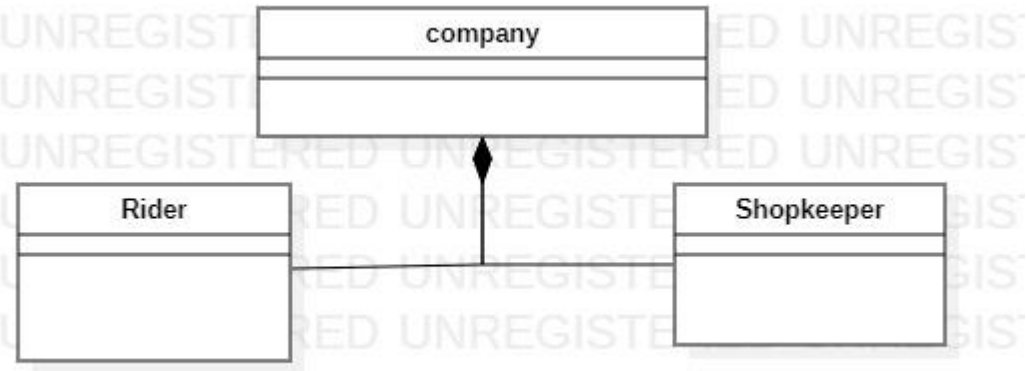
Object Oriented Features:

Composition:

The composition in OOP is a concept in which the class has the objects of the other classes Without which they cannot exist.

For Example:

In this project the composition relation is in the admin class which means whenever the object of the admin will have the rider object and the client object.



Inheritance:

Inheritance in OOP occurs when a class derives from another class. The child class will inherit all the public and protected properties and methods from the parent class. In addition, it can have its own properties and methods.

In this project we don't have any inheritance between classes.

Multiple Inheritance:

Multiple inheritance is a feature of some object-oriented computer programming languages in which an object or class can inherit features from more than one parent object or parent class.

IN this project we don't have Multiple Inheritance between classes.

Multi-Level Inheritance:

The multi-level inheritance includes the involvement of at least two or more than two classes. One class inherits the features from a parent class and the newly created sub-class becomes the base class for another new class.

IN this project we don't have Multilevel Inheritance between classes.

Polymorphism:

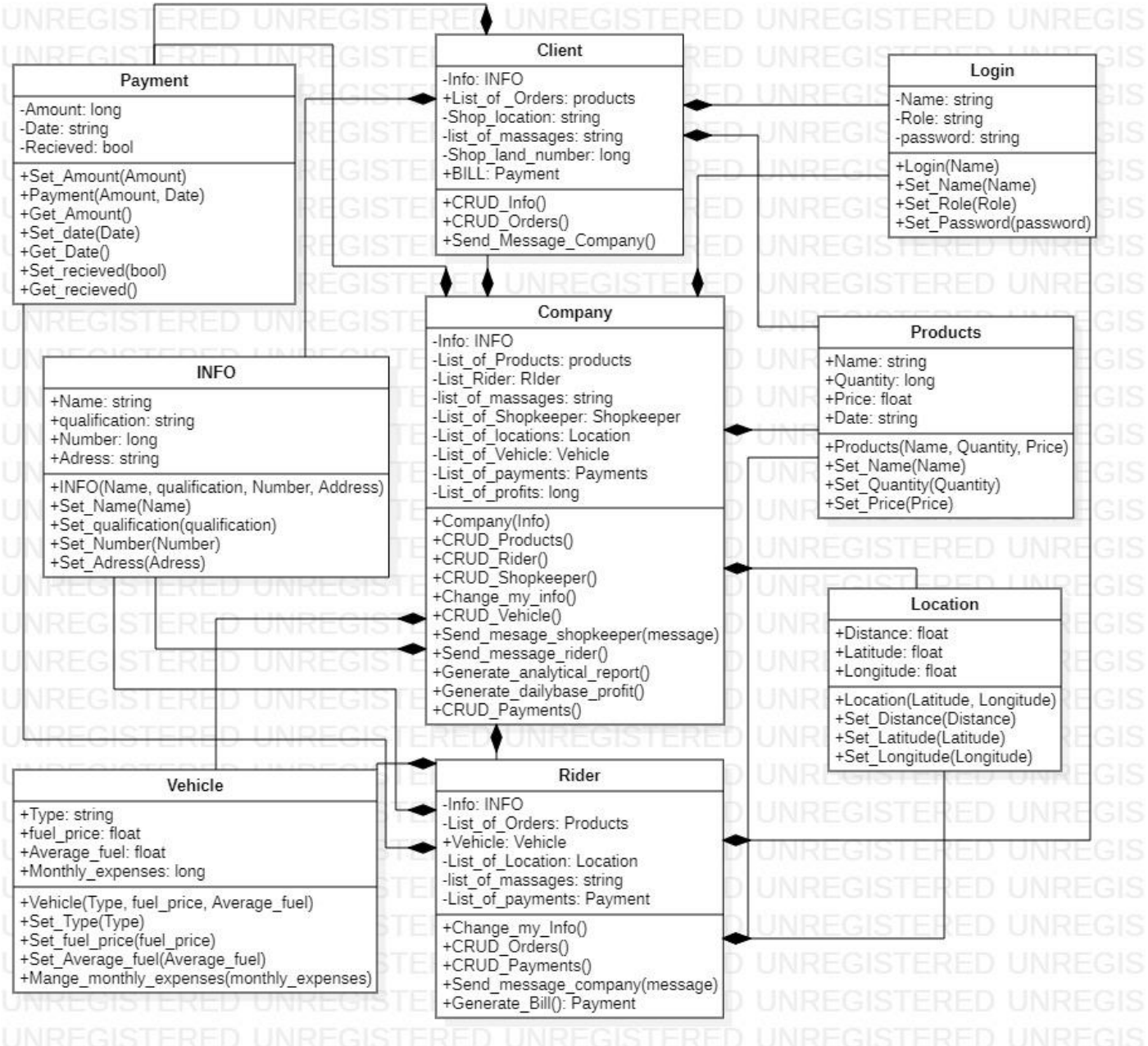
It is one of the four majors of the OOP which states that you can access the objects of different forms if they are same. For example:

A Ferrari, Bugatti and other cars are same because they all are 4-wheel vehicles.

IN this project we don't have Polymorphism between classes.

Detailed Object Oriented Design:

Draw complete design of project in StartUML.



Data Structure:

In this section, identity the use case in which you will use Data Structures e.g. Array List, LinkedList, Queue, Stack, HashSet and Tree Set etc. why you are forced to use these data structures.

Use Case Id	Data Structures Used	Justification for the usage of data structure
U01	No data structure.	Splash screen therefore no data structure used.
U02	Hash Algorithm	Hash function used to encrypt the password.
U03	Array List	Array helps to store values separately
U04	No Data Structure.	No need to store data.
U05	No Data Structure.	No need to store data.
U06	LinkedList	All data will used to save in the linked list.
U07	LinkedList	All data will used to save in the linked list.
U08	LinkedList	All data will used to save in the linked list.
U09	LinkedList	All data will used to save in the linked list.
U10	LinkedList	All data will used to save in the linked list.

Exceptions:

In this sections, identify at high level which type of exceptions you can face in your code and what are the solutions. Add more rows in the table as per requirements.

Type of Exception	Why this exception will occur	Use Case Id in which exception could be occurred	How you will handle the exception
Loading from file Exception	While loading different data type		The exception will be handled by reading files in arrays
Null exception thrown	If the object is null and we try to use it		The exception will be handled by exception handling
Assigning different data type to a variable	If we assign a value to a variable that is not of its data type		By taking value as string and then converting back to it
Stack overflow exception	By calling more than enough times a recursive call function		By setting the recursive call to minimum

Data Storage:

The files to read data from and write in will be of type "**.csv**" or simple "**.txt**".

1. Products.csv
2. Profits.csv
3. Logindetails.csv (Everyone)
4. InfoDetail_Employee.csv
5. Clients.csv (Info)
6. Rider.csv (Info)

7. Vehicle_details.csv
8. Payment_deatils.csv

Email Sending:

The email sending will occur in the following classes:

1. Email from the company/supplier to the deliverer
2. Email from the client to the company/supplier and deliverer
3. Email from the deliverer to the company/supplier

Project Plan

This section should include the implementation plan and work division among the members. All the estimated dates should be before December 20, 2022 including report and presentation.

Use Case Id	**Use Case Name**	**Member Name**	**Estimated Completion Date**
U1-U7	Splash Screen, Login, Registration, Admin Panel, Admin for Shopkeeper, Add Shopkeeper, Edit Shopkeeper	Hassan Wasti	17-12-2022
U8-U14	Delete Shopkeeper, Add Rider, Delete Rider, Edit Rider, Shopkeeper order Detail, Assign order to the rider for delivery, Product List	Bilal Qureshi	17-12-2022
U15-U21	Check Reviews, Location of Shop, Order taking Screen, Message interface for shopkeeper for rider, Complete or Incomplete, Shopkeeper stock, Shopkeeper reviews.	Afaq Ahmad	17-12-2022

Analytical Reports

In this section, you are required to describe the type of reports that are required by the management team of the distribution company. Also attach the format of the business reports.

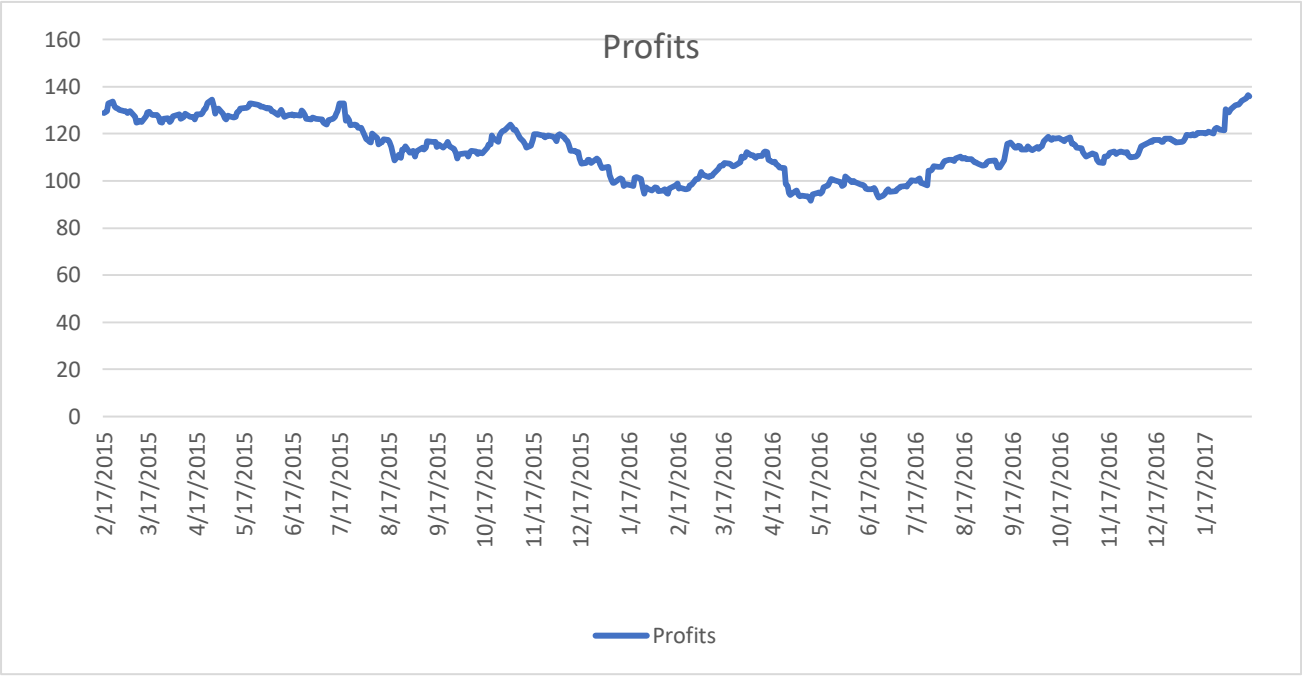


Fig: 1.1 Profit Details Everyday

Fig: 1.1 has a graph of the profits generated everyday by the company. The profits are measured every day and are are stored to see the growth of the company.

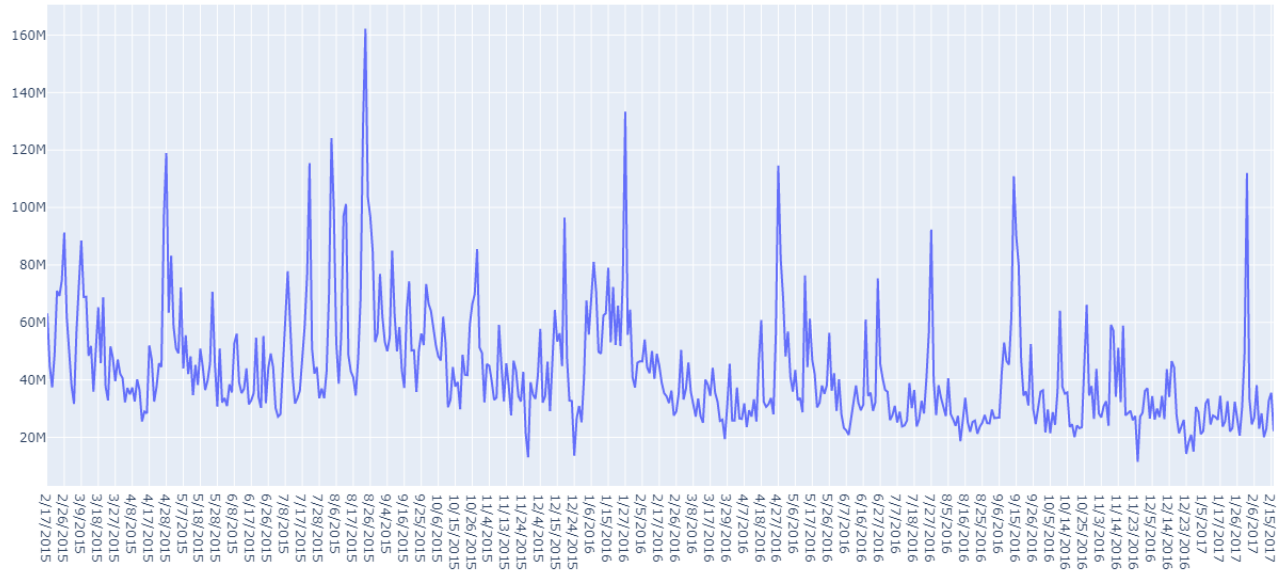


Fig: 1.2

Fig: 1.2 is the sales record of the company every day the stock order in the company to keep track of the intake and sale of the products.