Chapter 5

Analysis and Design

5.1 Activity Diagram:

5.1.1 Activity Diagram: Customer Registration
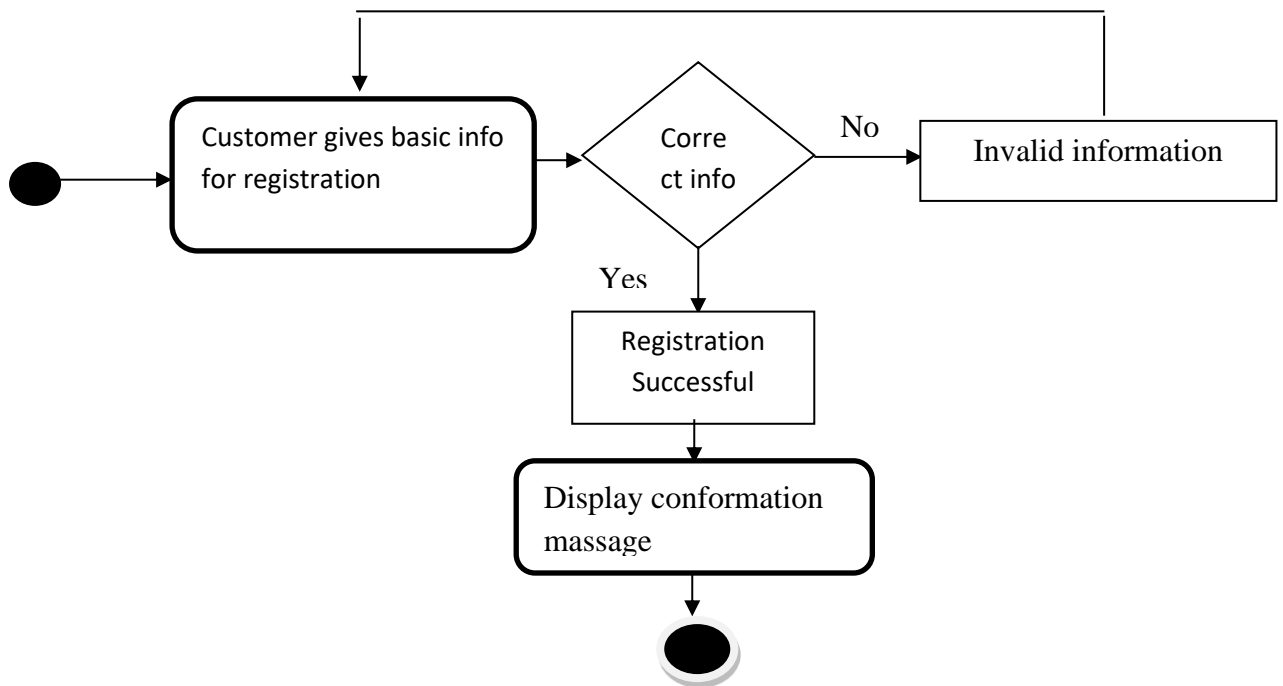

Figure 5.1:   Activity Diagram: Add User
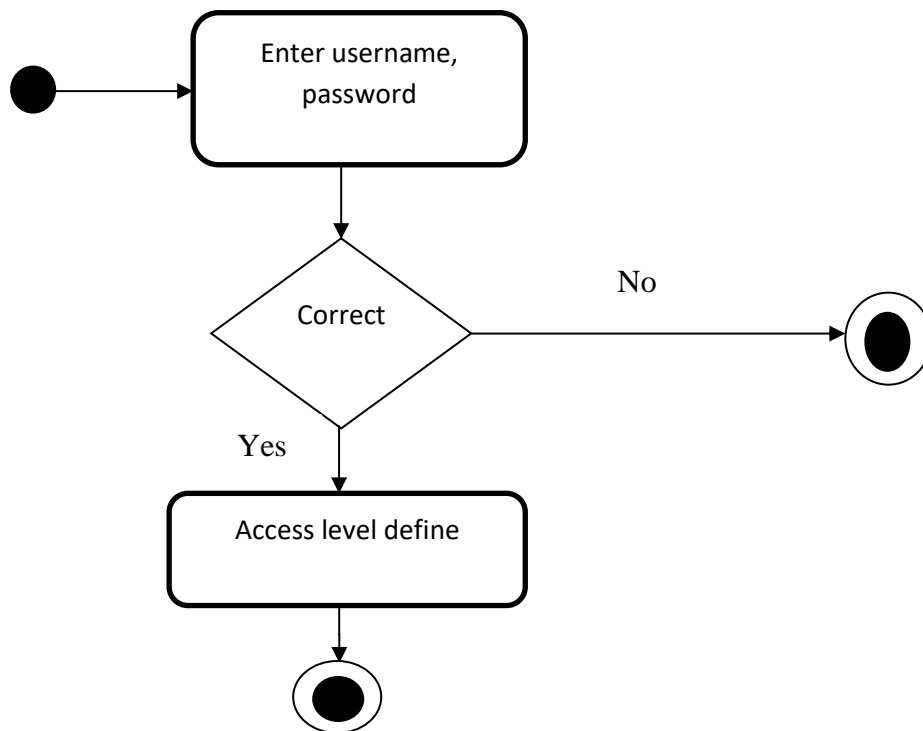
5.1.2 Activity Diagram: Activity Diagram: Login

Figure 5.2:   Activity Diagram: Login

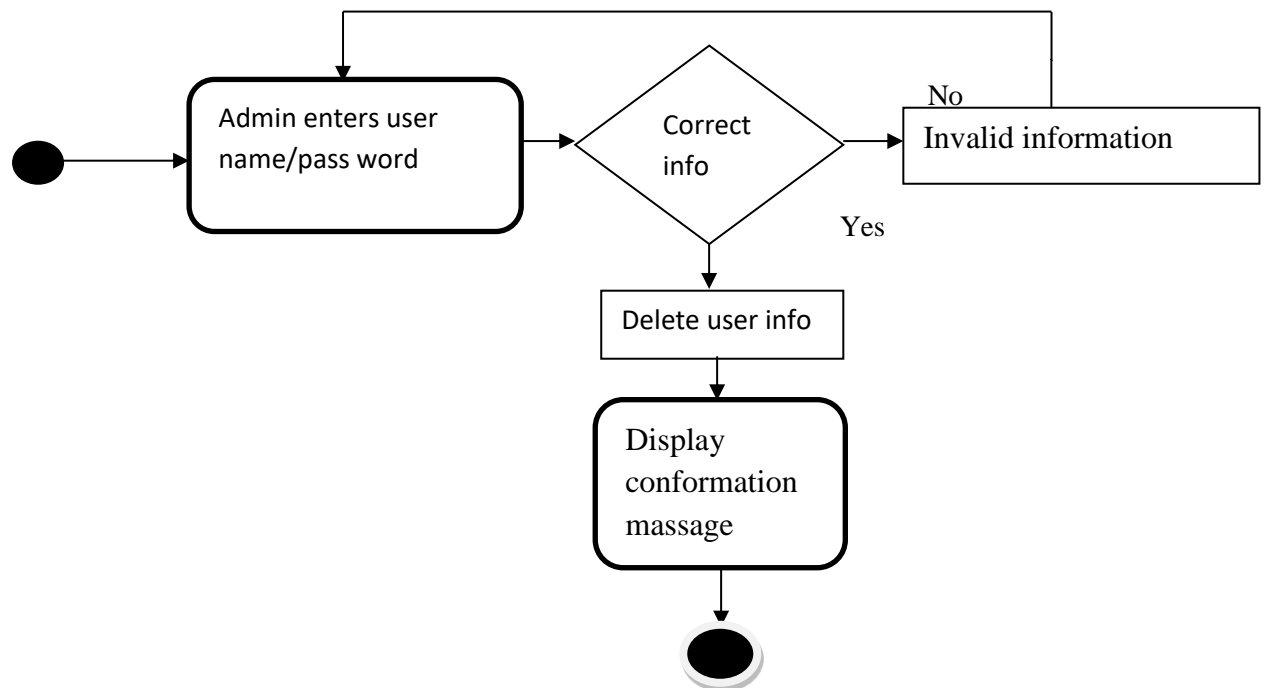### 5.1.3   Activity Diagram: Delete User

Figure 5.3:   Activity Diagram: Delete User

5.1.4   Activity Diagram: Add products
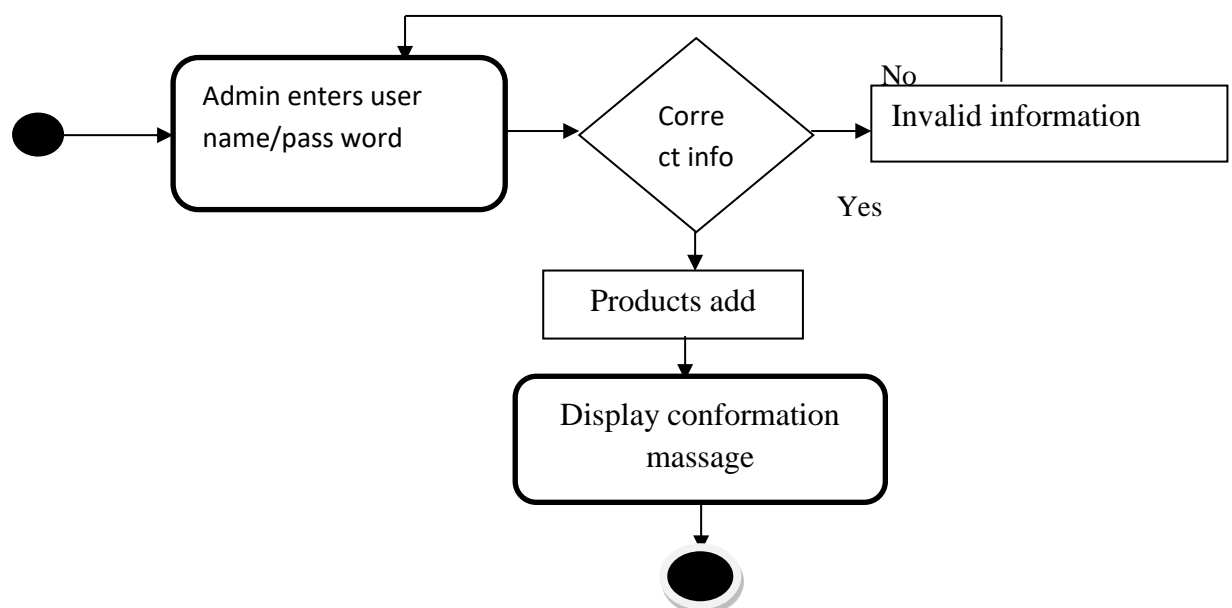

Figure 5.4:   Activity Diagram: Add products

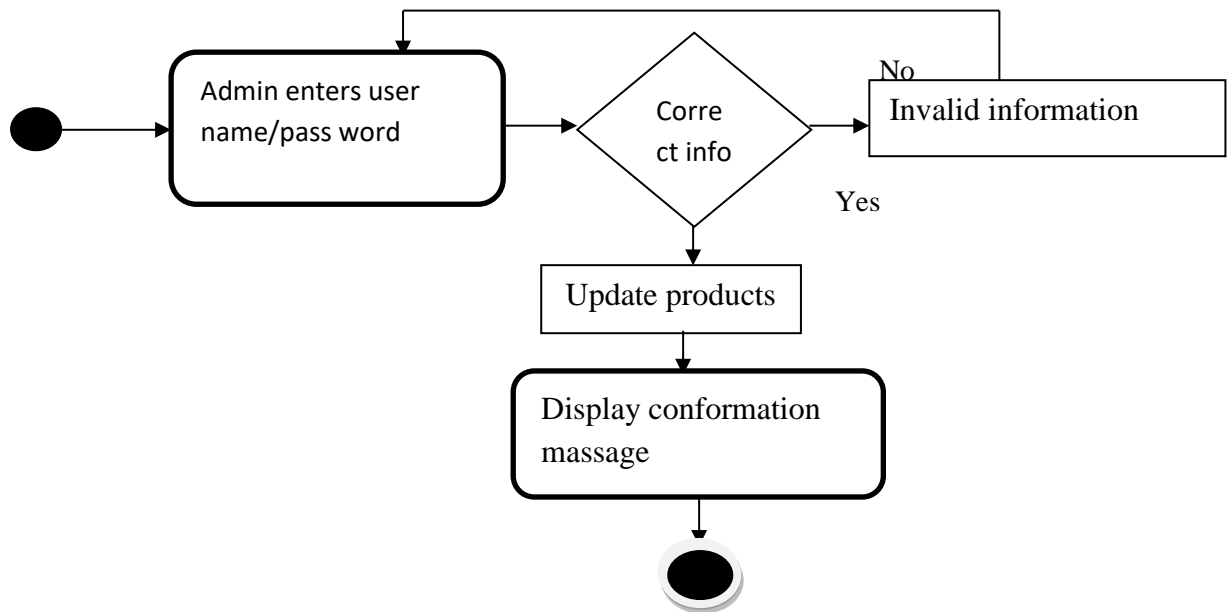### 5.1.6 Activity Diagram: Update products

Figure 5.5:   Activity Diagram: Update products

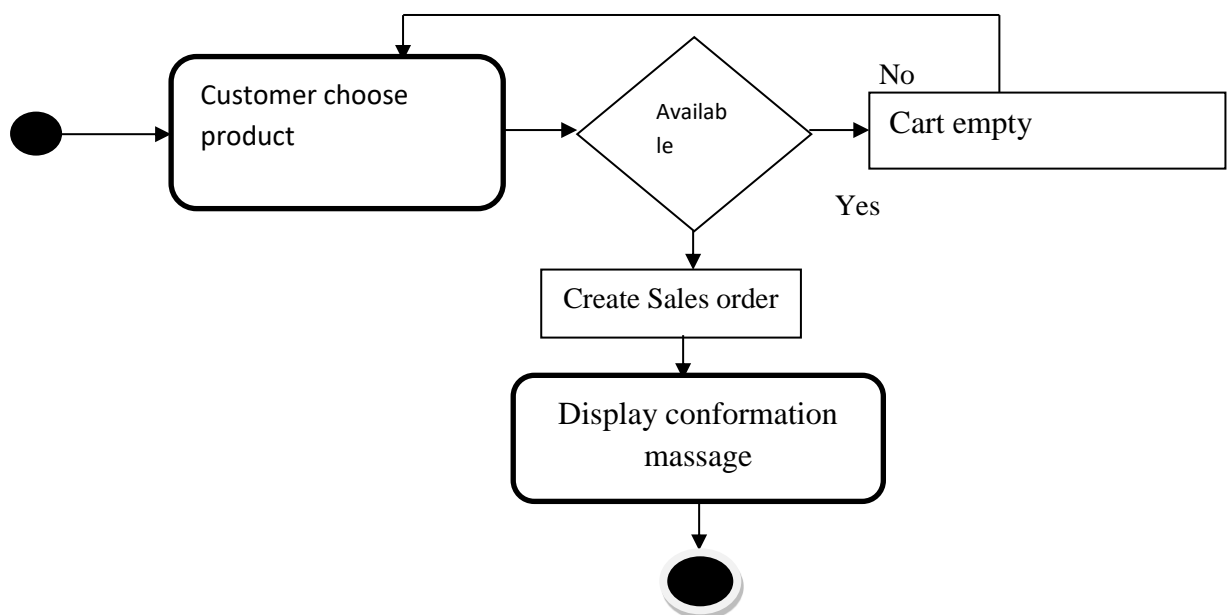### 5.1.7 Activity Diagram: Create Sales Order

Figure 5.6:   Activity Diagram: Create Sales Order

5.1.8 Activity Diagram: Update Sales Order
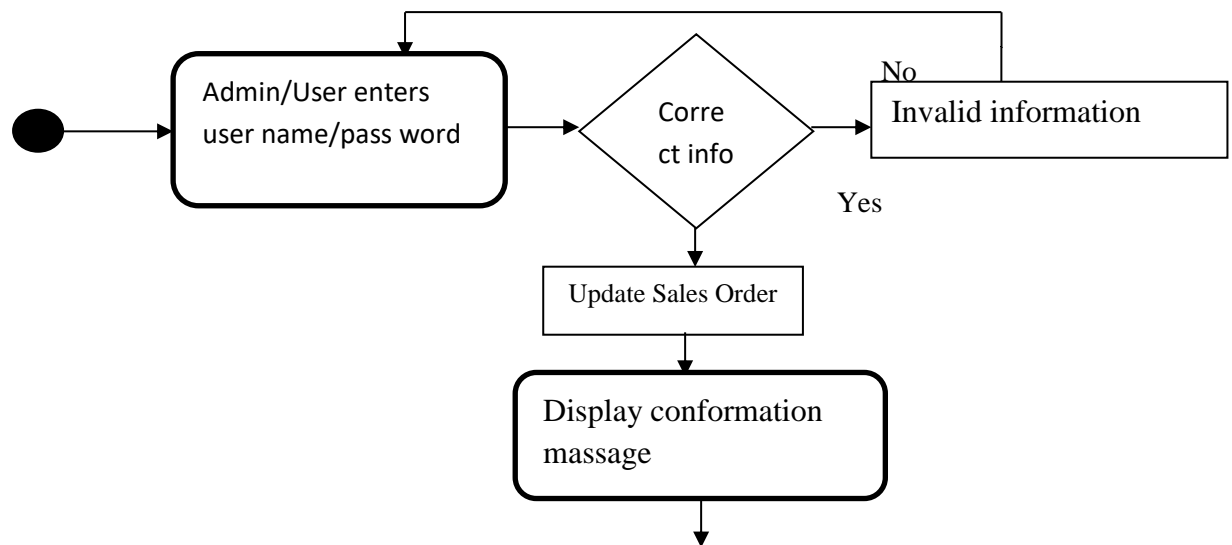

Figure 5.7:   Activity Diagram: Update Sales Order
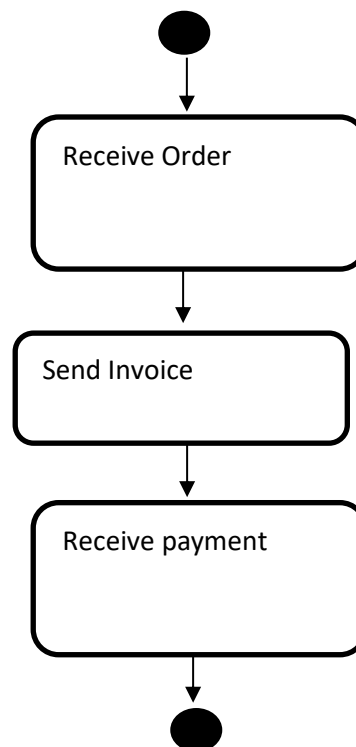
5.1.9 Activity Diagram: Payment

Figure 5.8: Activity Diagram for Payment

5.2   Class Diagrams

Class diagrams are the backbone of almost every object-oriented method including UML. They describe the static structure of a system.
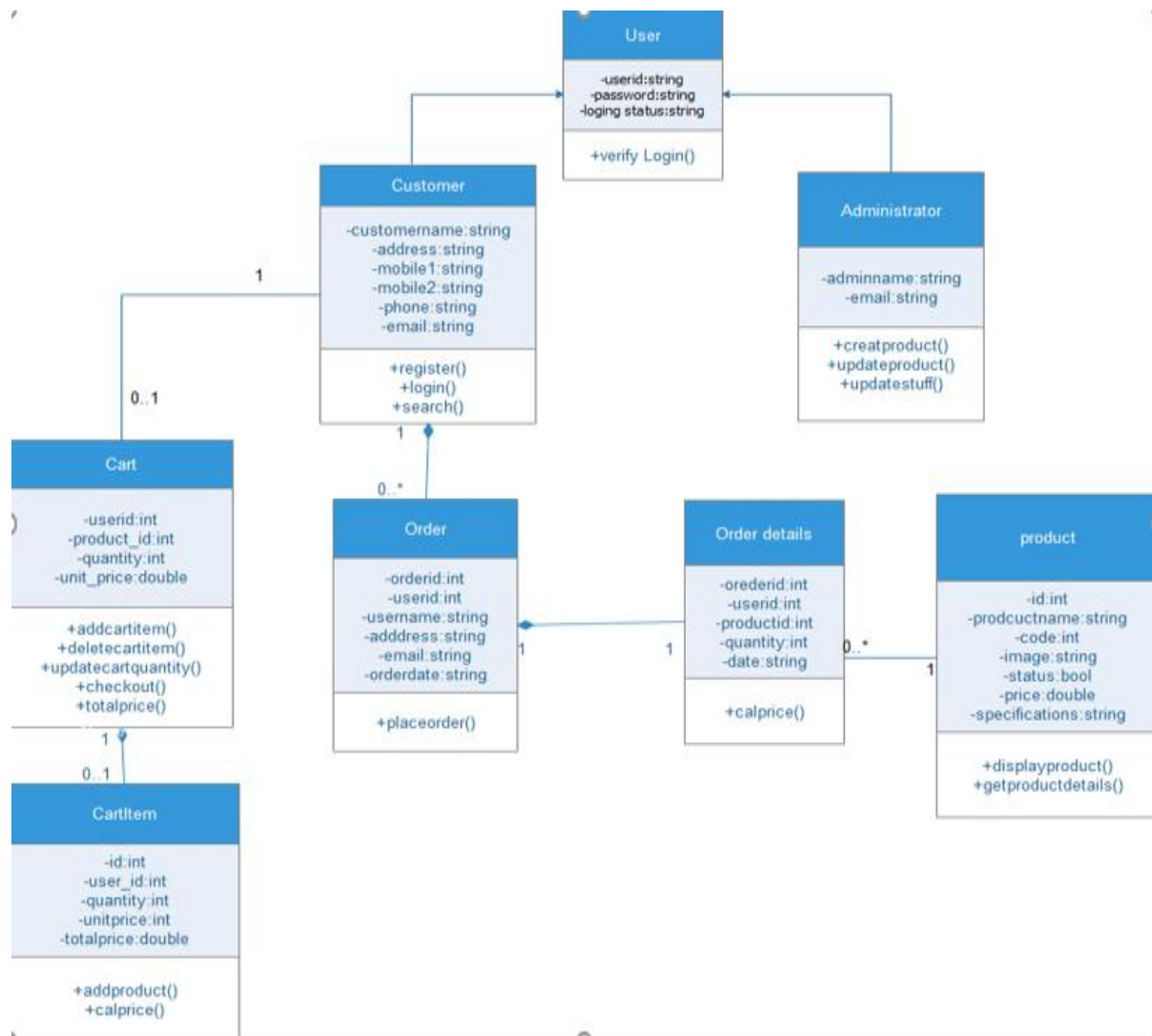

Figure 5.9: Class Diagrams

5.2.1 Class Descriptions

1) **User**

This class handles all user actions. The User class is the super class of Customer and Administrator. It includes the private methods to verify the login of the user. The verifyLogin() method is called when the user clicks the "Sign In" button. It returns true if the login is successful, false if it is not.

2) **Administrator**

This class handles the Administrator actions. It inherits all the User class responsibilities and its functions. It has a functions createproduct(), update product(), update stuff() are called when the administrator has to create/add a new department or category or product respectively to the catalog.

Conditions for createProduct Operation

- Purpose: To enable Administrator to create and add new products to the catalog.

- Pre Condition: Administrator must be logged in to be able to create a new product. Also, the department and/or category to which the new new product is to be associated should exist in catalog.

- Post Condition: Product will be added to the corresponding department and/or category.

- Input: Administrator will enter the name and necessary details to create a new product and click "Add" button to complete the action.

- Output: After the action, the changes to the catalog will be updated and saved and a message will be displayed accordingly.

3) **Customer**

This class handles the customer actions. It inherits all the User class responsibilities and its functions. customer must register and login through the Register and Login Webpage which uses the register() and login() methods. Also, the search() function is called when customer searches for some product and enters some Key-Words in the text-box on Search page.

Conditions for Login Operation

- Purpose: This is implemented to enable user authentication. A valid user account must be used for an existing customer.

- Pre Condition: The customer should be registered with website and should have a valid userid and password.

- Post Condition: Customer will be able to browse through the website.

- Input: The customer can login to the e-Commerce shopping system by entering his user name and password.

- Output: The system will verify that the login name matches the login password. If the user name or password is invalid, the appropriate error message will be indicated and the user will be requested to re-enter user name and password. If the user inputs are valid, the main page will be displayed.
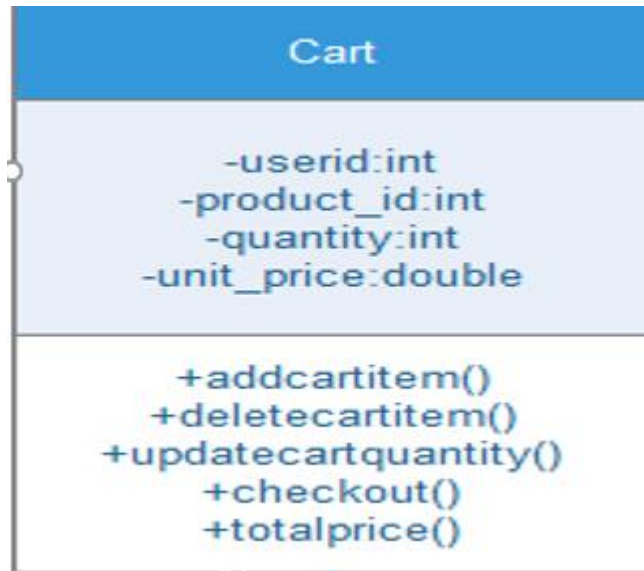
4) **Product**

This class represents collection of products of a particular category and/or department. The displayProduct() and getProductDetails() method are called when the user clicks on a product or on the "Search" button on the Search. Web form. The displayProduct()

method is used to retrive image of the product and getProductDetails() method retrives its details as a data set whenever a customer clicks on the product. A product can belong to one or more departments and/or categories.

5) **ShoppingCart**

ShoppingCart class has all the products that are added by a customer to buy. The addCartItem() and deleteCartItem() methods are called when customer performs actions like "Add to Cart" and "Delete" of products on the ShoppingCart. Webpage. The updatequantity() method is called when Customer clicks "Update" button on ShoppingCart. Webpage to increase or decrease the number of products in the cart. Also, viewCartDetails() method is called when customer clicks on the "View Details" button to see a summary of the cart. It will display the summary only when cart is not empty, else it will display "Your Cart is empty."

Conditions for addCartItem Operation

- Purpose: This is implemented so the customer can add products to shopping cart while searching or browsing catalog.

- Pre Condition: The customer must be logged in to add a product to the cart.

- Post Condition: Product will be added to the shopping cart.

- Input: When customer finds the products he wants, he adds them to the shopping cart by clicking on the "Add to Cart" button.

- Output: Product will be added to the shopping cart and the system will store and keep track of the information of products that have been added into shopping cart.

Conditions for Checkout Operation

- Purpose: To allow customer to buy the products added to the shopping cart.

- Pre Condition: Customer must be logged in and must have atleast one item in shopping Cart to be able to checkout and place the order.

- Post Condition: Customer will be succesfully checked out and will be able to edit his profile information and place order.

- Input: When the customer finishes shopping, he requests to checkout by clicking "checkout" button on Cart. page.

- Output: If the payment information of this customer already exists, the system prompts the customer to review or input a new one. If the credit card is valid, the order form will be processed by the system and checkout is complete.

6) Orders
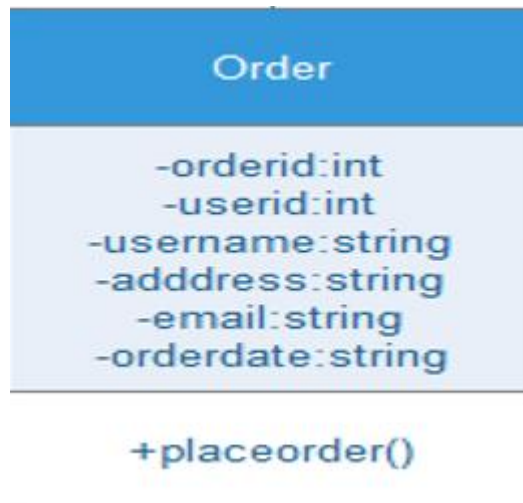
This class will store all information regarding the orders made by each customer. The placeOrder() method is called when customer clicks on the "Place Order" button on the Order Web form. It returns true if the order is placed successfully, false if it is not. The "Place Order" button will be enabled only when customer has a valid shopping cart and has entered valid personal, billing and   shipping   details.

Conditions for placeOrder Operation

- Purpose: To allow customer to place an Order for buying the products added to the shopping cart.
- Pre Condition: Customer must be logged in and must have atleast one item in shopping Cart to be able to place the order Also, customer should have correct profile information and valid Credit Card details entered into the system.
- Post Condition: Customer will be succesfully able to place the order.
- Input: When the customer finishes shopping, he requests to place an order by clicking "Place Order" button on Order page.
- Output: If the profile information, payment information the credit card of this customer is valid, the order form will be processed by the system and order is placed.

7) OrderDetail

Order details

-orederid:int
-userid:int
-productid:int
-quantity:int
-date:string

+calprice()

This Class handles details regarding every order that the customer makes. The calcPrice() method is called to calculate the total amount of the order placed. The method also calculates the total amount of the order including the shipping charges once the customer has selected the shipping region.

8) CartItem

CartItem

-id:int
-user_id:int
-quantity:int
-unitprice:int
-totalprice:double

+addproduct()
+calprice()
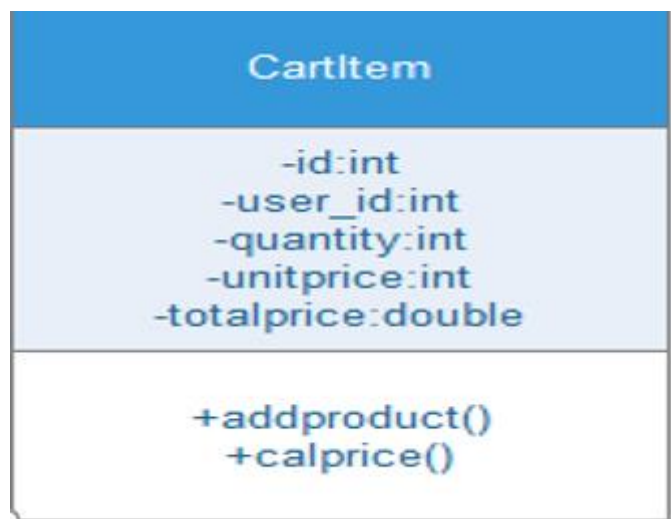
A CartItem object indicates the quantity, unitcost, subtotal amount of the product selected by a customer. When a customer performs a product selection and clicks "Add to Cart" button, a new cartItem object is created and added to the cart. The calcPrice() method is called to calculate the total amount of the items added to cart.

5.3   Entity Relationship Diagram(ERD)

An entity-relationship diagram (ERD) is a data modeling technique that graphically illustrates an information system's entities and the relationships between those entities. An ERD is a conceptual and representational model of data used to represent the entity framework infrastructure.

The elements of an ERD are:
- Entities
- Relationships
- Attributes

Steps involved in creating an ERD include:

1. Identifying and defining the entities
2. Determining all interactions between the entities
3. Analyzing the nature of interactions/determining the cardinality of the relationships
4. Creating the ERD

Entity

Entities are represented by means of rectangles. Rectangles are named with the entity set they   represent.

An entity represented by a rectangle.

Attributes

Attributes are properties of entities. Attributes are represented by means of eclipses. Every eclipse represents one attribute and is directly connected to its entity (rectangle).

A relationship described by aneclipse.

Relationship

Relationships are represented by diamond shaped box. Name of the relationship is written in the diamond-box. All entities (rectangles), participating in relationship, are connected to it by a line.
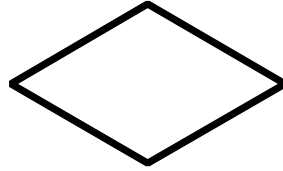
A relationship described by a diamond.

BINARY RELATIONSHIP AND CARDINALITY

A relationship where two entities are participating, is called a binary relationship. Cardinality is the number of instance of an entity from a relation that can be associated with the relation

One-to-one

When only one instance of entity is associated with the relationship, it is marked as '1'. This image below reflects that only 1 instance of each entity should be associated with the relationship. It depicts one-to-one relationship
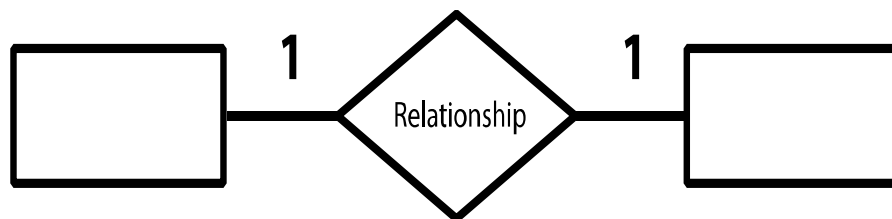
One-to-many

When more than one instance of entity is associated with the relationship, it is marked as 'N'. This image below reflects that only 1 instance of entity on the left and more than one instance of entity on the right can be associated with the relationship. It depicts one-to-many relationship
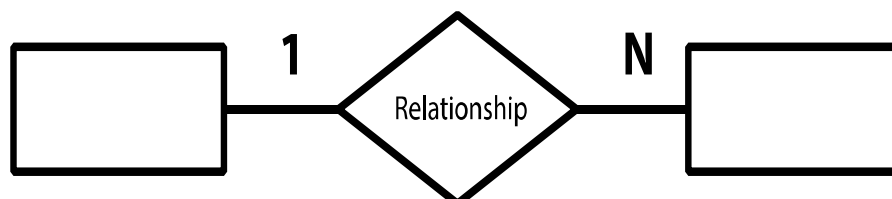
Many-to-one

When more than one instance of entity is associated with the relationship, it is marked as 'N'. This image below reflects that more than one instance of entity on the left and only one instance of entity on the right can be associated with the relationship. It depicts many-to-one relationship.
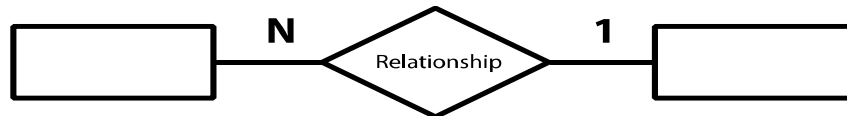
Many-to-many

This image below reflects that more than one instance of entity on the left and more than one instance of entity on the right can be associated with the relationship. It depicts many-to-many relationship
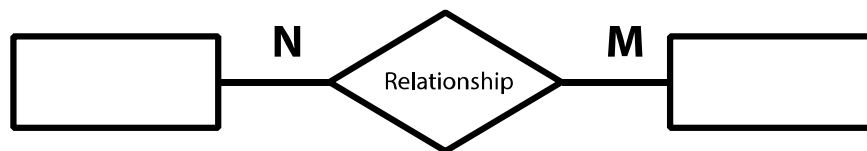
PARTICIPATION CONSTRAINTS

Total Participation: Each entity in the entity is involved in the relationship. Total participation is represented by double lines.

Partial participation: Not all entities are involved in the relationship. Partial participation is represented by single line.
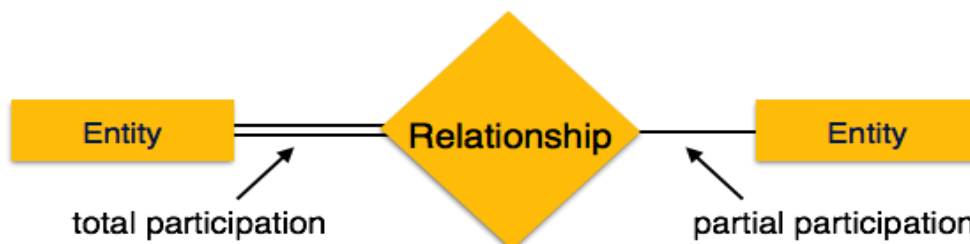
Primary Key: A primary key is an attribute or collection of attributes that allow us to identify an entity uniquely.

Foreign key: A foreign key is an attribute of a relation, which refers to an existing attribute of another relationship.

ER Diagram
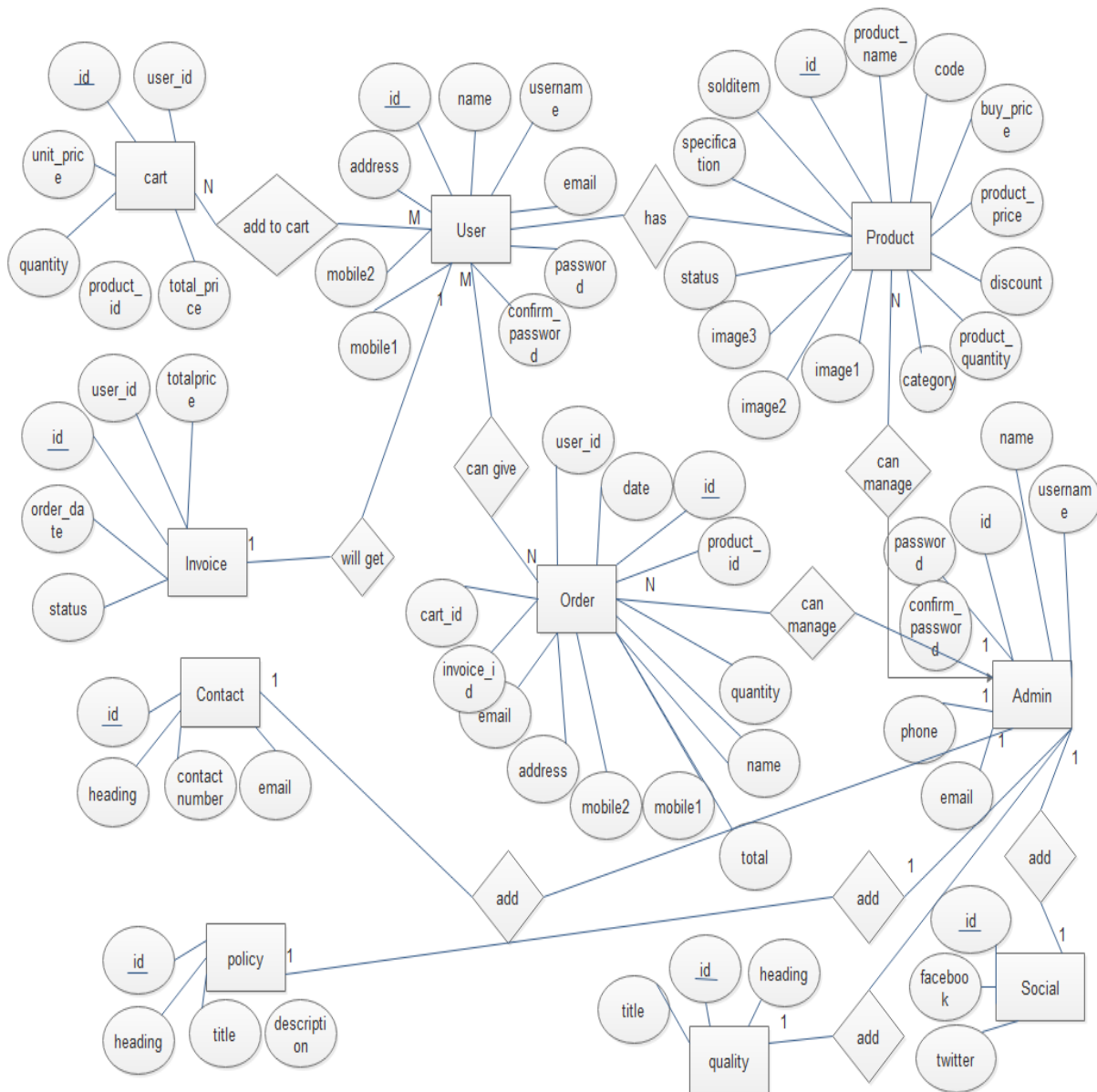
Figure 5.10: Entry Relationship Diagram

5.4 Data Flow Diagram

A data flow diagram is a graphical representation that depicts information flow and the transforms that are applied as data move from input to output.

The DFD may use to represent a system or software at any level of abstraction. DFD may partition into levels that represent increasing information flow and functional detail. Therefore, the DFD provides a mechanism for functional modeling as well as information flow modeling.

A level 0 DFD, which is also known as fundamental system model or a context model, represents the entire software or system element into as a single bubble with input and output data indicated by incoming and outgoing arrows respectively. Then bubble of context model should decompose into several levels.

5.4.1 Data Flow Diagram of E-Commerce System:
5.4.1.1 Context Level Diagram
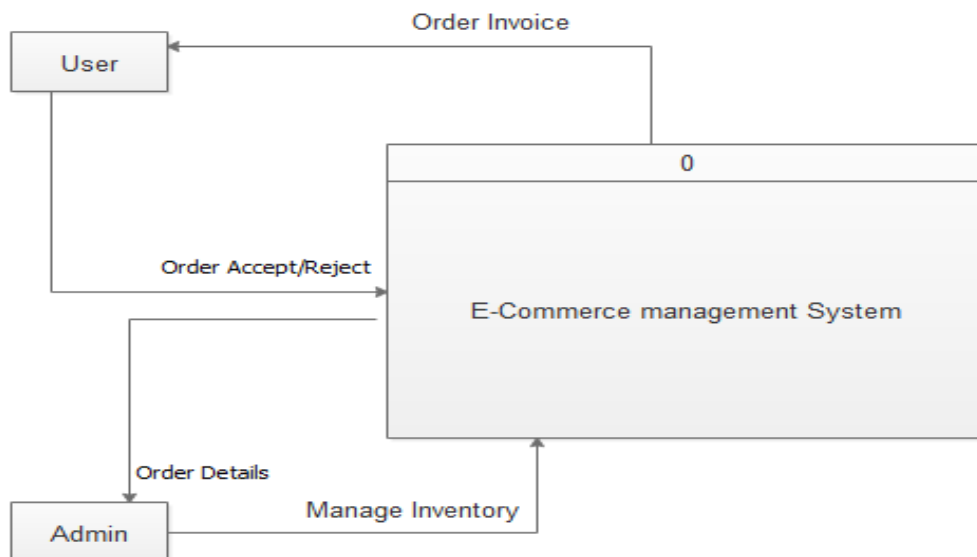
Figure: 5.11: Context Level Diagrams

5.4.1.2 Level 1 Diagram:
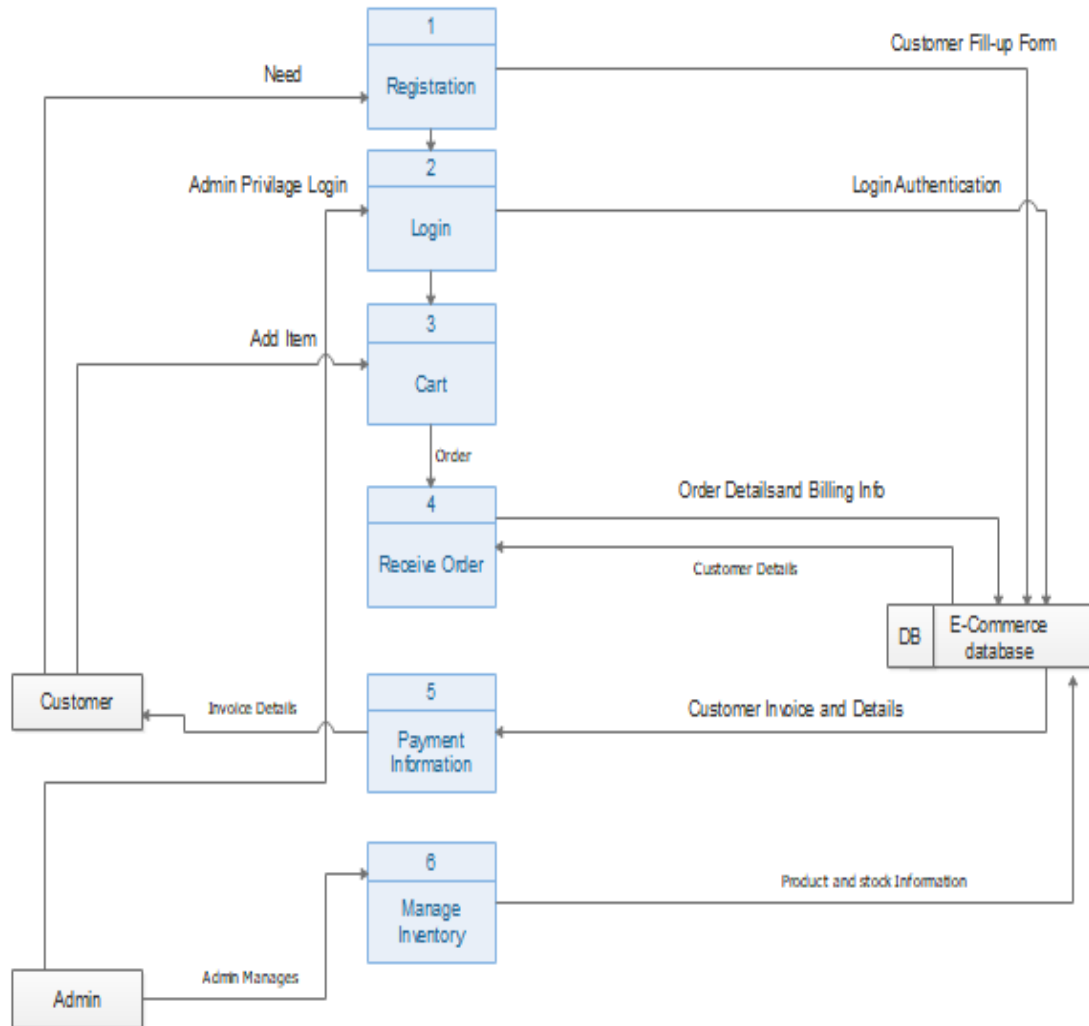
Figure 5.12: Level 1 Diagrams

5.4.1.3 Level 2 Process 1 Diagram (Registration):
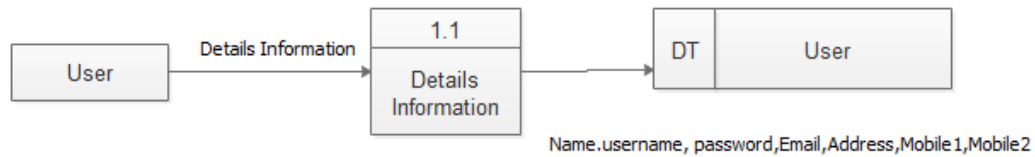
Figure 5.13: Level 2 of Process 1 Diagrams

5.4.1.4 Level 2 Process 2 Diagrams (Login):
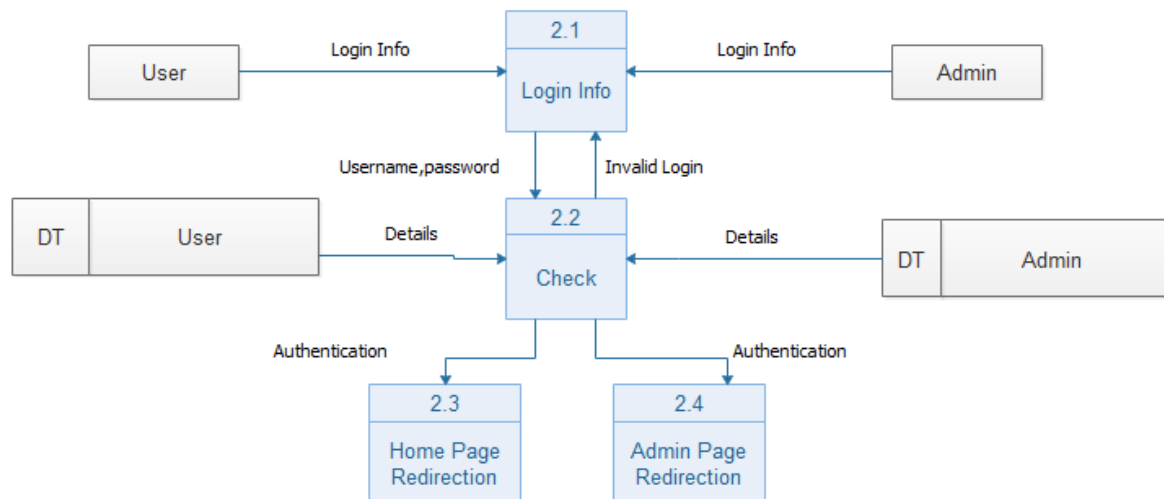
Figure 5.14: Level 2 of Process 2 Diagrams

5.4.1.5 Level 2 Process 3 Diagrams(Cart):
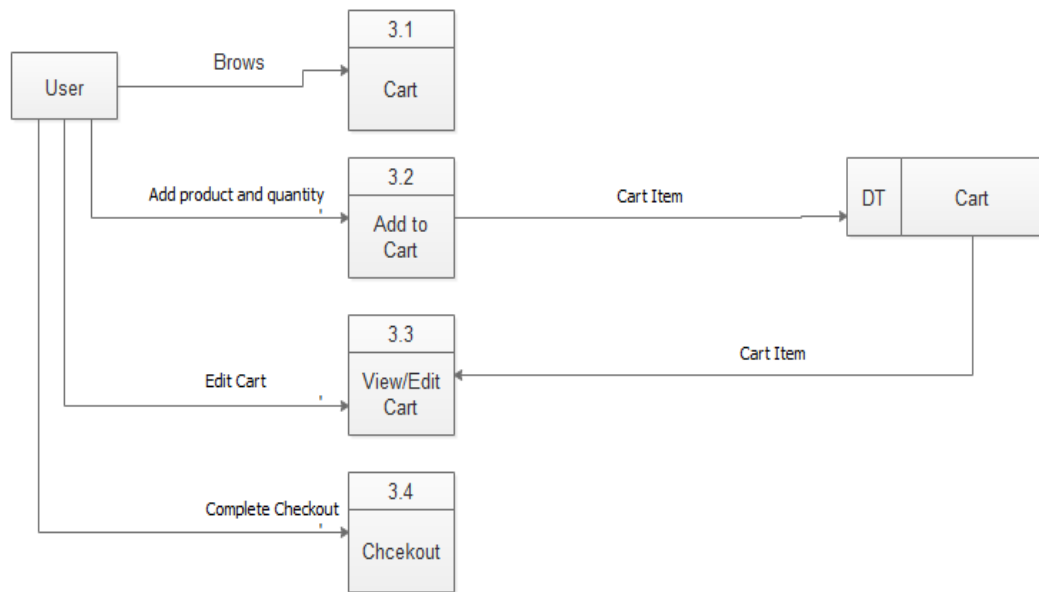
Figure 5.16: Level 2 of Process 3 Diagrams

5.4.1.6 Level 2 Process 4 Diagrams(Order):
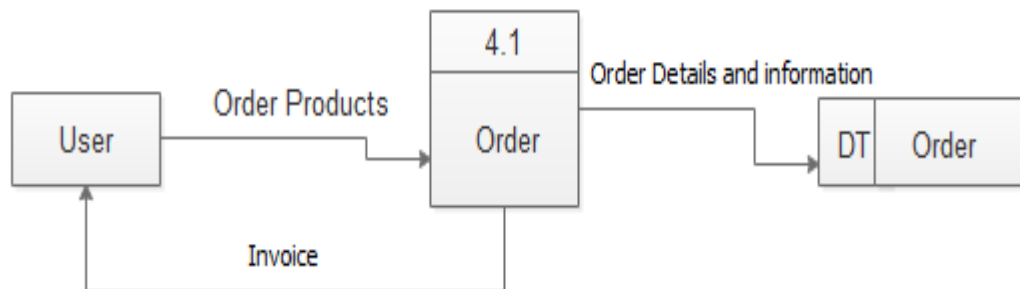
Figure 5.17: Level 2 of Process 4 Diagrams

5.4.1.7 Level 2 Process 5 Diagrams(Payment):
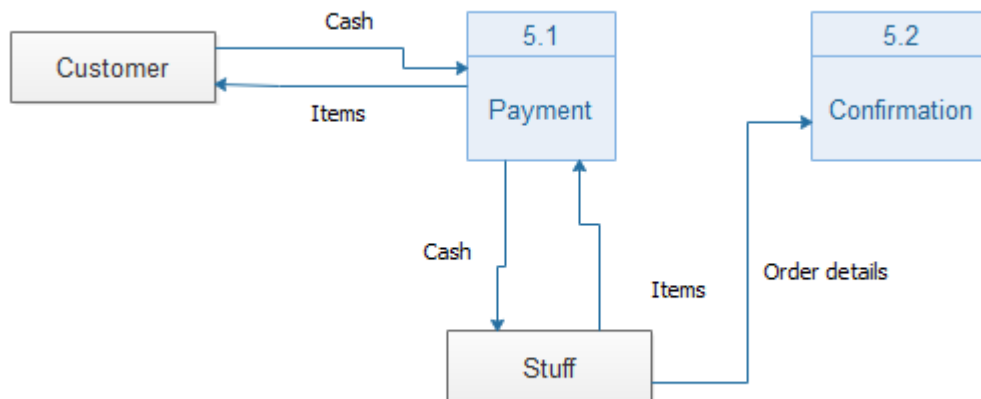
Figure 5.18: Level 2 of Process 5 Diagrams

5.4.1.8 Level 2 Process 6 Diagrams (Stock):
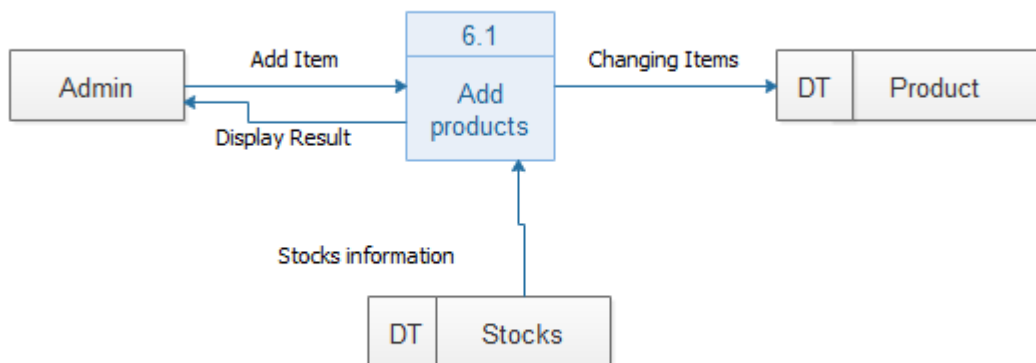
Figure 5.19: Level 2 of Process 6 Diagrams

5.5 Swim Lane Diagram:
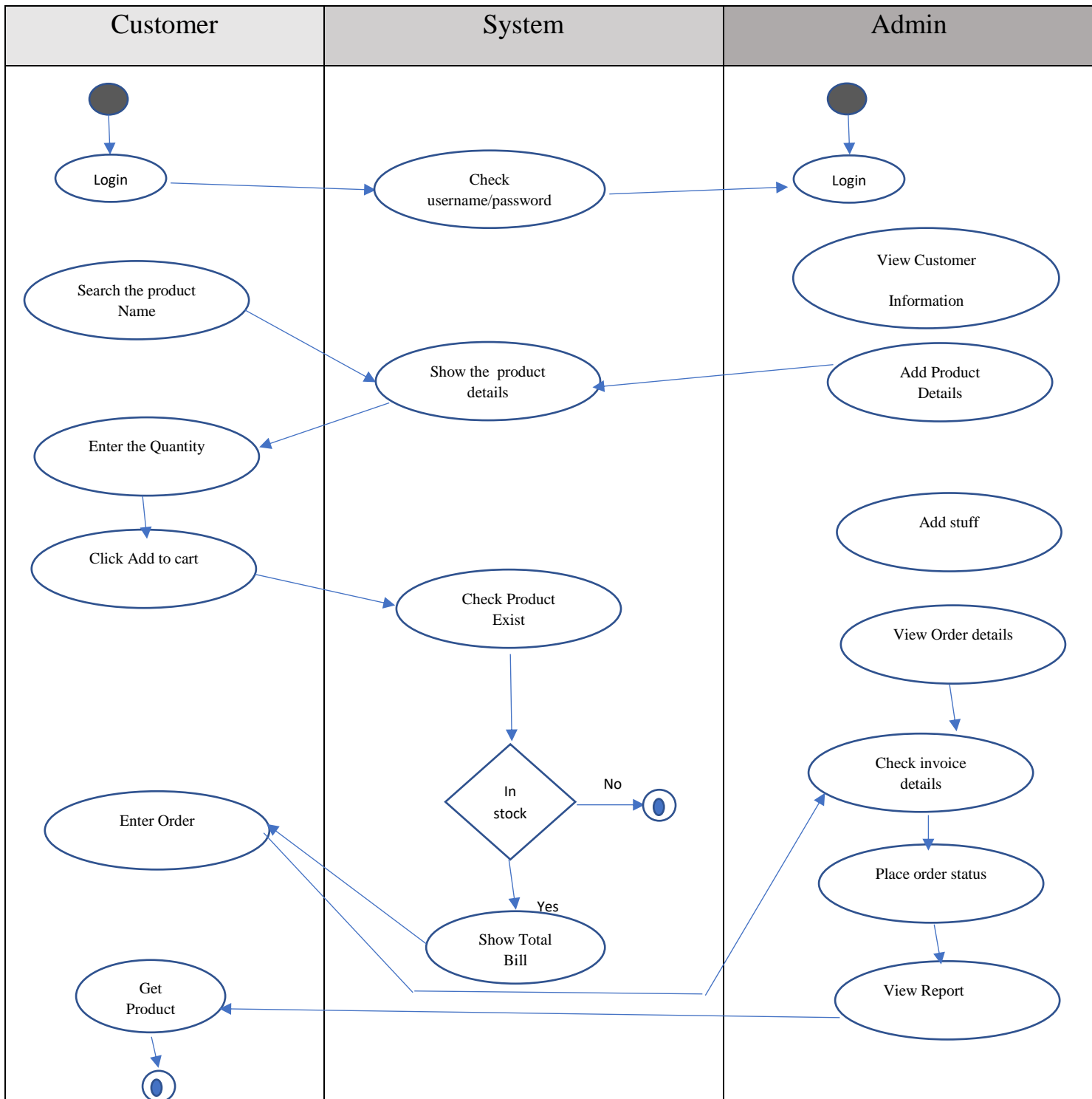
Figure 5.20: Swim Lane Diagram

5.6 Interface:

5.6.1: Login Page:

Figure 5.21: Admin Login Page

5.6.2: Dash Board:
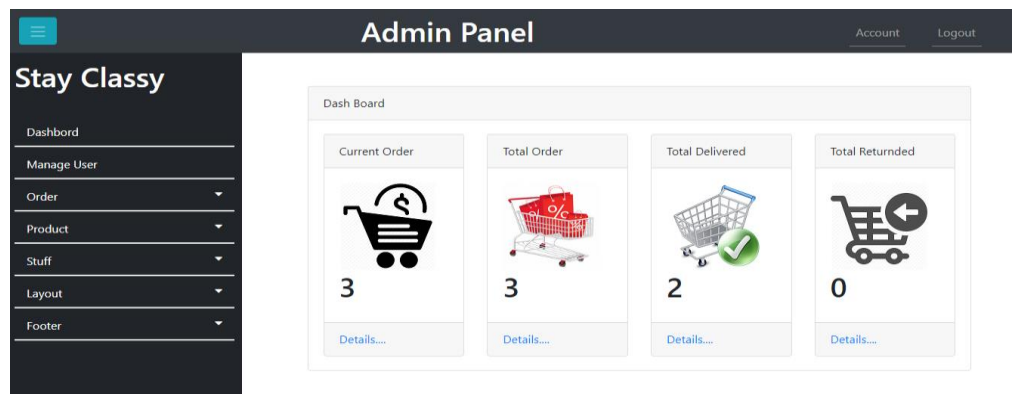
Figure 5.22 Admin Dash Board

5.6.3: Add Product:

Figure 5.23: Add Product

5.6.3: Order:

All Order

Customer Name	Mobile 1	Address	Order Date	Status	Price	Details
Arifuzzaman	12121212	Uttora Dhaka	2018-12-10	delivered	49650	Show
Ifat	32423423	Dhaka	2018-12-09	delivered	46200	Show
Ifat hasan	121212	uttora	2018-12-09	pending	35200	Show

Figure 5.24: Order

5.6.4: View Product:

All Products

Image	Name	Code	Category	Type	Buy Price	Price	Discount %	Quantity
	Office Bag	Office-1	Travelling	Gents	900	1100		79 Add
	Gents Bag	Gents-1	Travelling	Gents	800	1000		192 Add
	Travelling Bag	Travelling-1	Travelling	Gents	1800	2500		-59 Add

Categories

☐ Travelling
☐ Duffel
☐ Office
☐ Regular
☐ Other bags

Category Types

☐ Gents
☐ Ladies

Figure 5.25: View Product

5.6.5: View stuff:

Stuff List

Id	Name	Username	Phone	E-Mail	Delete
1	Amiruzzaman	Admin	01787284068	munna.ak17@gmail.com	Delete
3	Munna Amir	Amir Admin	1641064685	munna.ak17@gmail.com	Delete

Figure 5.26: View Stuff

5.6.6: View Transection:
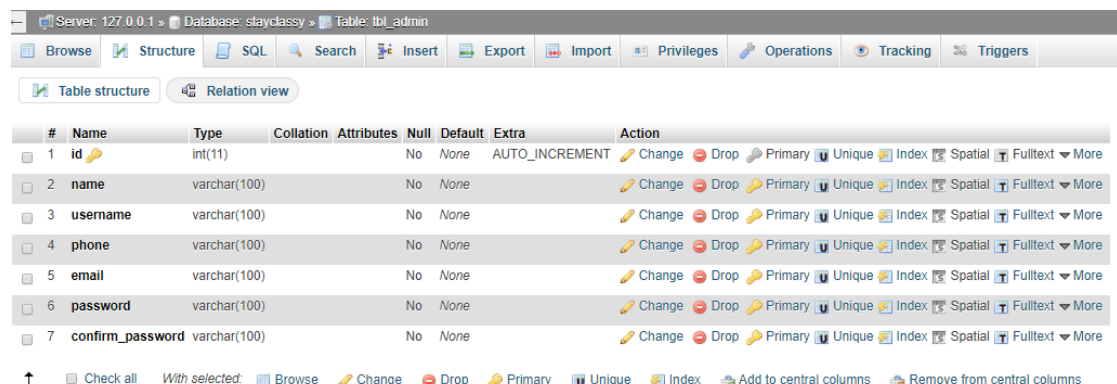
Transection History

Date:	TO:	Submit	Amount	Role
Total			Invest=0	Income=0
				Loss = 0

Download

Figure 5.27 : Transection History

5.7 Database Design:

Database, often referred as DB is an essential part of software. There are various database platforms like MySQL, SQL Server, Oracle Database, MongoDB and so on. In my software, I have used MySQL as it is freeware, user friendly and can be run on local machine.

5.7.1 Table for "Admin":

Figure 5.28 : Admin table

5.7.2 Table for "Product":

Figure 5.29 : Product table

5.7.3 Table for "Order":

Figure 5.30 : Order table

5.7.4 Table for "Transection":

Figure 5.31 : Transection table