

A decorative border resembling a scroll, with a vertical strip on the left and horizontal lines at the top and bottom. The scroll is light gray with a thin black outline. The corners are rounded, and there are small circular details at the top and bottom of the vertical strip.

# **CHAPTER-3**

## **DESIGN OF THE PROJECT**

## **3.Design of the Project**

### **3.1 System Design**

#### **3.1.1 Introduction**

System design is the process or art of defining the architecture, components, and modules. Interface and the data for a system to satisfy specified requirements. System design is also called as top-level design, in which the focus is on deciding which modules is need for the system.

#### **3.1.2 Overview**

System design is therefore the process of defining and developing a system to satisfy specified requirements of the user. Until the 1990's standardization of hardware and software resulted in the ability to build modular systems. The increasing the importance of the software running on the generic platform has enhanced the discipline of software engineering.

Object oriented analysis and design methods are becoming the most widely standard language used in the object-oriented analysis and design. It is widely used for modeling software system and it is increasingly used for big designing non software system and organizations.

#### **3.1.3 Logical Design**

The logical design of the system pertains to an abstract representation of data flows, inputs, and outputs of the system. This is often conducted via modeling, which involves a simplistic representation of an actual system. In the context of system design, modeling can undertake the following form, includes

- Data flow diagrams
- Entity relationship diagram

### **3.1.4 Physical Design**

The physical design relates to actual input and output processes of the system. This is laid down in terms of how data is input the system, how it is verified. Physical design, in this context, does not refer to the tangible physical design of an information system.

To use an analogy, a personal computers physical design involves input via keyboard. Processing within the CPU, and output via monitor, printer etc. It would not concern the actual layout of the tangible hardware, which for a pc would be a monitor, CPU, motherboard, hard drive, modems, video/graphics cards, USB slots etc.

### **3.1.5 Scope and Overview**

#### **3.1.5.1 Scope**

The scope of the project is from giving the constraint and accessing the data and represents it in graphical form. It helps the user to understand the data. To do this we find the solution using collected requirements and design this solution. Design is done in this phase. This phase is done in correct way because it effects the next phases also.

#### **3.1.5.2 Overview**

The software requirement specification step of a software development process yields specifications that are used in software engineering. If the system is semi-automated or user-cantered, software design may involve user experience design yielding a story board to help determine those specifications.

If the software is completely automated, a software design may be as a flowchart or text describing a planned sequence of events. There are also semi standard method like Unified Modelling Language and Fundamental modeling concepts. In either case some documentation of the plan is usually the product of the design. A software design may be platform independent or platform specific, depending on the availability of the technology called for by the design.

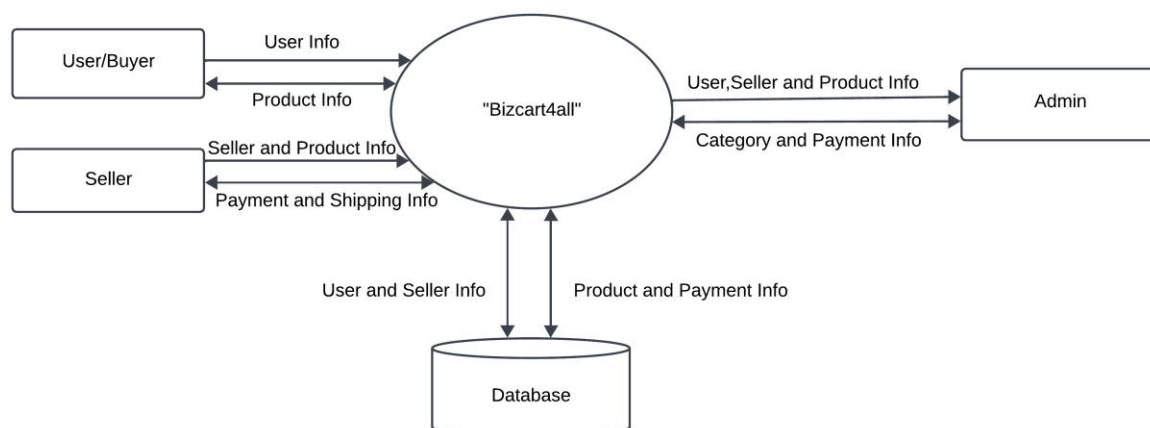
## 3.2 System Overview

Mining the user information and represent it in the graphical form is the main functionality of a software. We develop an application which working on the concept of data mining and satisfies the user constraints.

### 3.2.1 Context Flow Datagram (CFD)

A context flow diagram defines the boundary between the system, or part of the system, and its environment, showing the entities that react with it. It represents all the external entities that may interact with the system. In context flow diagram, the entire system is treated as a single process and all input, output, sinks and source of the process are shown. Using CFD Design methodology DFD of the system is represented. The environment in which (the context) the software is used is depicted in CFD.

The CFD shows the external entity acting on the software. The CFD shows the external entity on the software. The software is shown here in CFD as a single process. The best CFD's are used to display how a system interoperates at a very high level, or how systems operate and interact logically.



**Fig.3.2.1.1 Control Flow Diagram**

### 3.2.2 Data Flow Diagram (DFD)

Data flow diagram is also called Data Flow Graph or Bubble chart which is commonly used during problem analysis. A DFD shows the flow of data through a system as a function that transforms the inputs into desired outputs.

The process in DFD is represented by named circle and data flow are represented by named arrows entering or leaving the bubbles. A source or sink is typically outside the main system of study. The two main purposes of DFD are:

- To provide an indication of flow of data as they move through the system.
- To depict the function that transforms the flow.


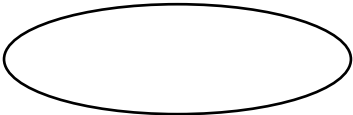
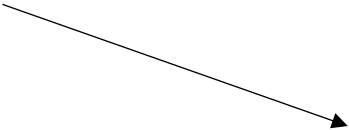
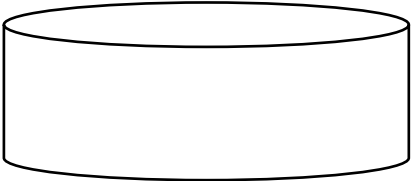
The DFD provides additional information that is used during the analysis of the problem domain and serves as a basis for modelling functions. The DFD may be used to represent a system or software at any level of abstraction. In fact, DFD provides a mechanism for functional modelling as well as information flow modelling.

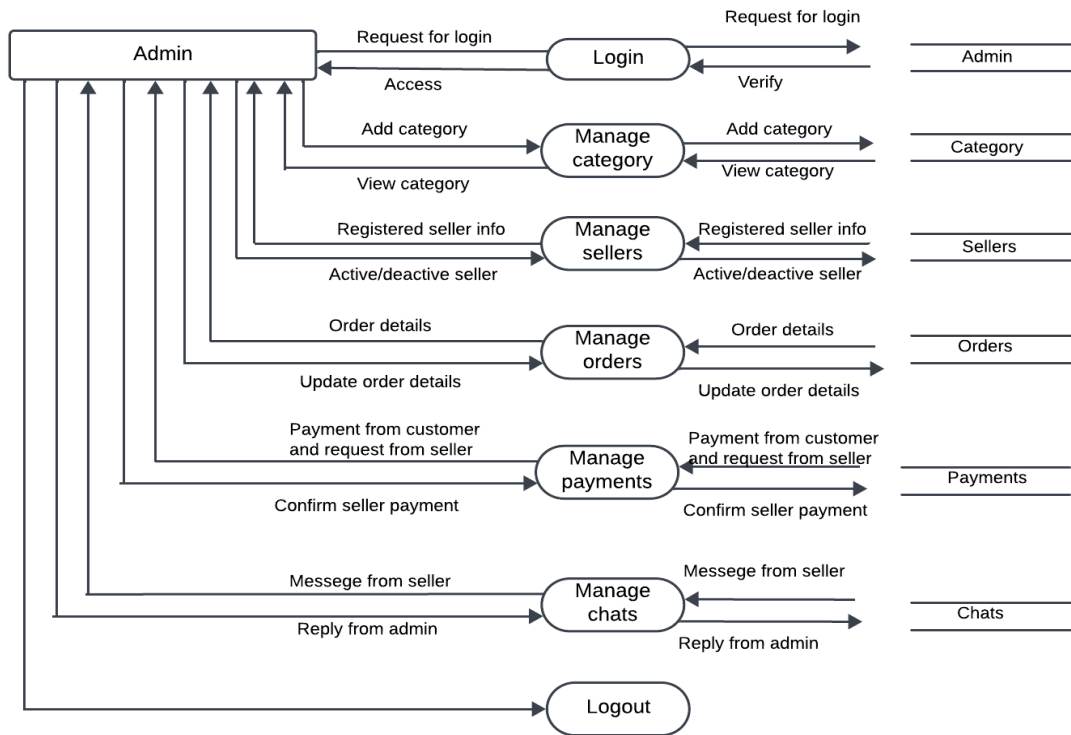
A level 0 DFD also called fundamental system model or a context model representing the entire software elements as a single bubble with inputs and output data indicated incoming and outgoing arrows respectively. Additional process and information flows are represented as level 0 DFD, which is partitioned to reveal more details. For example: a level 0 DFD might contain four or five bubbles with interconnecting arrows. Each of the processes represented level 1 is a sub function of overall system depicted in the context model.

The basic notation used to create a DFD makes it easy to analyse and understand. The DFD is a graphical tool that can be very valuable during software requirement analysis.

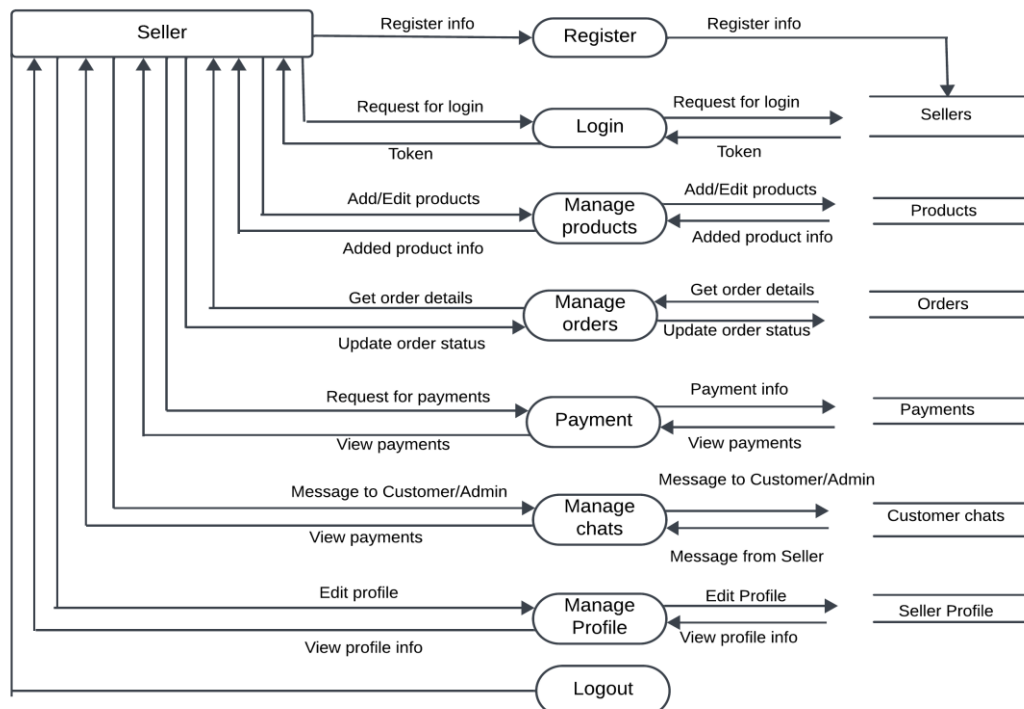
### **3.2.2.1 Notation for DFD**

**Table.3.2.2.1 Notation for DFD**

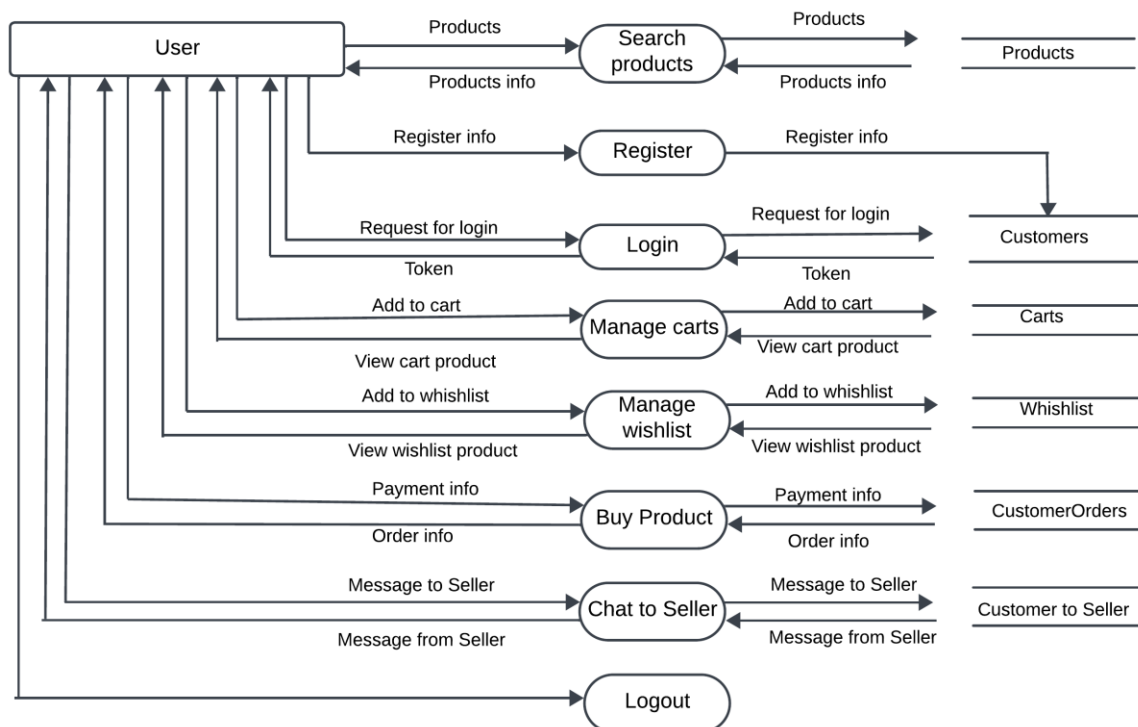
<b>ENTITY:</b> 	External entities are outside the system, but they either supply input data into the system or use system output. These represented by rectangle. It is used for specifying from data where data comes and where it reaches.
<b>PROCESS:</b> 	A process shows a transformation or manipulation of data flows within the system. A process transforms incoming data flow into outgoing data flow.
<b>DATAFLOW:</b> 	A data flow shows flow of information from source to destination. A data flow is represented by a line, with arrow head showing the direction of flow.
<b>DATABASE:</b> 	Databases are outside the system, is used to store the data in backend.
<b>TABLES:</b> _____ _____	Tables are the part of databases. Used to store the data.



**Fig.3.2.2.1 Admin Data Flow Diagram**



**Fig.3.2.2.2 Seller Data Flow Diagram**






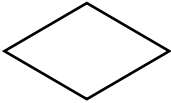
**Fig.3.2.2.3 User Data Flow Diagram**

### 3.2.3 Flow Chart

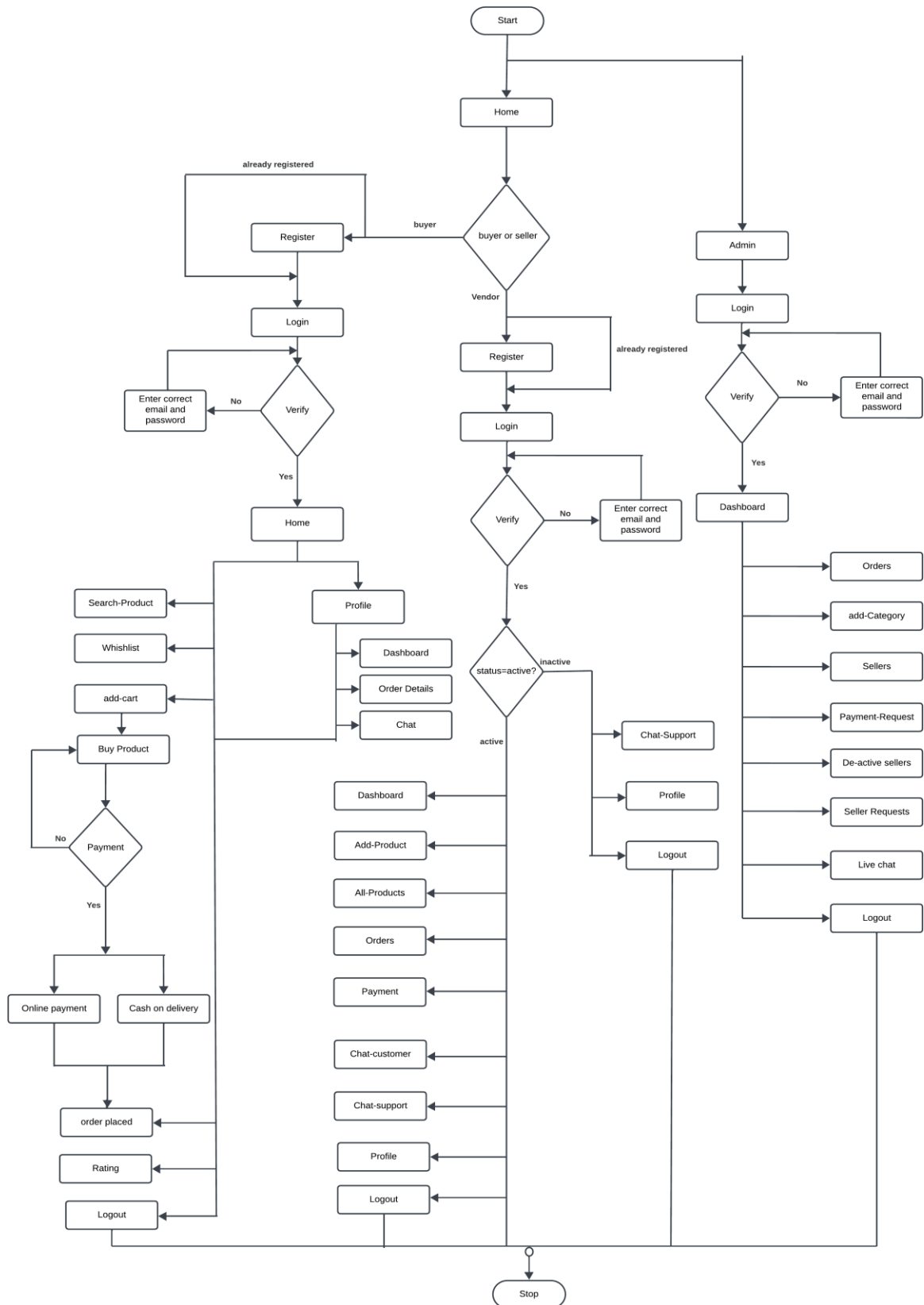
A flowchart is a type of diagram that represents a workflow or process. A flowchart can also be defined as a diagrammatic representation of an algorithm, a step-by-step approach to solving a task. A simple flowchart representing a process for dealing with a non-functioning lamp. The flowchart shows the steps as boxes of various kinds, and their order by connecting the boxes with arrows. This diagrammatic representation illustrates a solution model to a given problem. Flowcharts are used in analysing, designing, documenting or managing a process or program in various fields.

### Notation for Flowchart



<b>SYMBOL</b>	<b>PURPOSE</b>	<b>DESCRIPTION</b>
	Flow line	Used to indicate flow of logic by connecting symbols.
	Terminal(start/stop)	Used to represent start and end of flowchart.
	Processing	Used for arithmetic operations and data manipulation.
	Decision	Used to represent the operation in which there are two alternatives, true and false.

**Table.3.2.2.2 Notation for Flowchart**




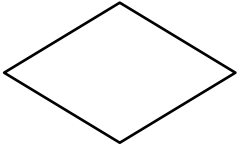
**Fig.3.2.2.4 Flowchart**

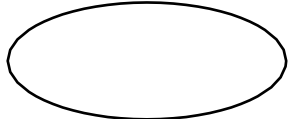
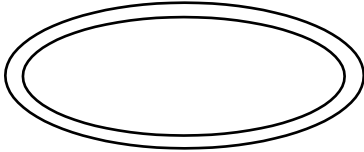
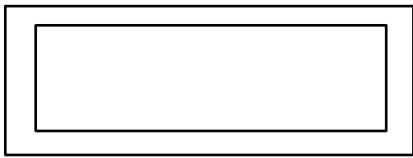
### 3.2.3 Entity Relationship (ER) Model

Entity Relationship model is popular high-level conceptual data model. This model its variations are frequently used for the conceptual design of the database applications many databases design tools employ its concepts. We described the basic data structuring concepts and constraints of the model and discuss their use in the design of conceptual; schemas for database application.

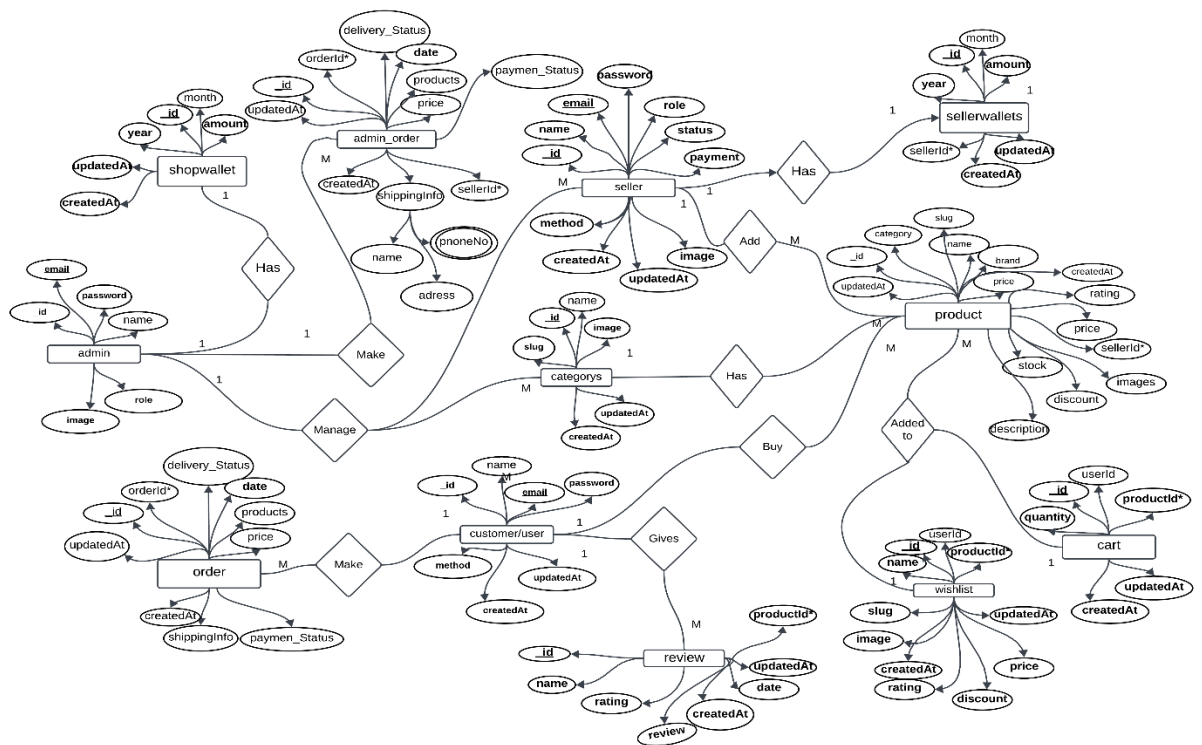
The basic units in an ER model are entities, their attributes, entity types and relationships between entity types.

- **Entities:** An entity represents an object defined within the information system about which you want to store information. An entity may be defined as a thing which is recognized as being capable of independent existence and which can be uniquely identified. An entity is an abstraction from the complexities of domain.
- **Entity Attributes:** Attributes are elementary pieces of information attached to entity. It is the entities where the number of attributes that can be used to describe them potentially vast, but the number that will actually apply to a given entity is relatively modest.
- **Relationships:** A relationship is a named connection or association between entities. A relationship captures how entities are related to one another.

Components	Description
	A rectangle or process symbol is generally used to represent entity.
	The diamond symbol is used to show relationships between entities.

	The terminator or oval symbol is used to define the attributes of the entity.
	The double oval symbol is used to define the multi valued attribute of the entity.
	Weak entity

**Table.3.2.3.1 Notation For E-R diagram**



**Fig.3.2.3.1 Entity Relationship Diagram**

## 3.3 Database Design

### 3.3.1 Data Object Design

Database design is required to manage the large bodies of information. The management of data involves both the definition of structure of the storage of information and provisions of mechanism for the manipulation of information. In addition to the dB system must provide for the safety of information handled, despite the system crashes due to attempts at unauthorized access. For developing an efficient dB, we will have to fulfil certain condition such as:

- Control Redundancy
- Ease of use
- Data independence
- Accuracy and integrity

There are 6 major steps in designing process.

- Identify the table and relationship
- Identify the data that is need for each table and relationship.
- Resolve the relationship
- Verify the design
- Implement the design

### 3.3.2 Tables

The BizCart4all-Multivendor E-commerce Web Application uses following tables as fields:

#### adminModel

Field	Type	Required	Default
adminname	String	Yes	-
email	String	Yes	-
password	String	Yes	-
role	String	No	admin
createdAt	Date	No	Date. now
updatedAt	Date	No	Date. now

**Table 3.3.2.1 Admin Model**

Field	Type	Required	Default
name	String	Yes	-
email	String	Yes	-
password	String	Yes	-

role	String	No	seller
status	String	No	pending
payment	String	No	inactive
method	String	Yes	-
image	String	No	‘
shopinfo	Object	No	{}
createdAt	Date	No	Date. now
updatedAt	Date	No	Date. now

#### **sellerModel**

**Table 3.3.2.2 Seller Model**

#### **customerModel**

Field	Type	Required	Default
name	String	Yes	-
email	String	Yes	-
password	String	Yes	-
createdAt	Date	No	Date.now
updatedAt	Date	No	Date.now

**Table 3.3.2.3 Customer Model**

#### **categoryModel**

Field	Type	Required	Default
name	String	Yes	-
image	String	Yes	-
slug	String	Yes	-
createdAt	Date	No	Date.now
updatedAt	Date	No	Date.now

**Table 3.3.2.4 Category Model**

#### **productModel**

Field	Type	Required	Default
sellerId	Schema.ObjectId	Yes	-
name	String	Yes	-
slug	String	Yes	-

category	String	Yes	-
brand	String	Yes	-
price	Number	Yes	-
stock	Number	Yes	-
discount	Number	Yes	-
description	String	Yes	-
shopName	String	Yes	-
images	Array	Yes	-
rating	Number	No	0
createdAt	Date	No	Date.now
updatedAt	Date	No	Date.now

**Table 3.3.2.5 Product Model**

### **wishlistModel**

Field	Type	Required	Default
userId	Schema.ObjectId	Yes	-
productId	String	Yes	-
name	String	Yes	-
price	Number	Yes	-
slug	String	Yes	-
discount	Number	Yes	-
image	String	Yes	-
rating	Number	No	0
createdAt	Date	No	Date.now
updatedAt	Date	No	Date.now

**Table 3.3.2.6 Wishlist Model**

### **cartModel**

Field	Type	Required	Default
userId	Schema.ObjectId	Yes	-
productId	Schema.ObjectId	Yes	-
quantity	Number	Yes	-
createdAt	Date	No	Date.now
updatedAt	Date	No	Date.now

**Table 3.3.2.7 Cart Model****stripeModel**

Field	Type	Required	Default
sellerId	Schema.ObjectId	Yes	-
stripeId	String	Yes	-
code	String	Yes	-
createdAt	Date	No	Date.now
updatedAt	Date	No	Date.now

**Table 3.3.2.8 Stripe Model****customerOrder**

Field	Type	Required	Default
customerId	Schema.ObjectId	Yes	-
products	Array	Yes	-
price	Number	Yes	-
payment_status	String	Yes	-
shippingInfo	Object	Yes	-
delivery_status	String	Yes	-
date	String	Yes	-
createdAt	Date	No	Date.now
updatedAt	Date	No	Date.now

**Table 3.3.2.9 CustomerOrder Model**