**ONLINE MOBILE AND LAPTOP SELLING**

**Submitted By:**

**1.Mahendrasinh Zala**

**2.Meet Patel**

**3.Hardik Desai**



External Guide Internal Guide

Zeel Thakkar Sanjay Soni

**Navgujarat College of Computer Applications**

**Ashram Road, Ahmedabad**

**December -2020**

## INDEX

1. PROJECT PROFILE.………...……………………………………..…...
2. COMPANY PROFILE……………………………………….………….
3. ANALYSIS
   1. Existing System…………………………………..…………………
   2. proposed System……….…………………….…...…………………
   3. Advantages of proposed system………………….……………….…
   4. 3.5.Tools&Technology…………………………………………………..
4. SYSTEM FLOW Diagram
   1. Symbols of System Flow Diagram …………..………………….….
   2. Admin System Flow Diagram……………...……………..…...........

User System Flow Diagram …….………………………….…….

4.4 Visitor System Flow Daigram……………………………...………

1. UML DIAGRAM
   1. USE CASE DIAGRAM
      1. Symbol of Use Case Diagram …………………...…...….
      2. Use Case Diagram
   2. ACTIVITY DIAGRAM
      1. Symbol of Activity Diagram …………………….……....

5.2.2. Login Activity ……………....……….…..

5.2.3. Registration Activity ……………………….…..………

5.2.4 .Insert Prodcuct Activity ……………..…………………….....

5.2.5 .Order Activity ……………..…………………………

* + 1. Cart Activity…………………………….………………
    2. Payment Activity…………………………...………………
  1. SEQUENCE DIAGRAM
     1. Symbol of Sequence Diagram ….………....….………….30
     2. Login Sequence ………….…………..….………...33
     3. Registration Sequence ………………........………....….…..…..34

5.3. 4 . Insert Product Sequence ………………………….….…........35

5.3.5 Order Sequence …………..………….…………….…......36

* + 1. Cart Sequence………………….….….………………..37
    2. Payment Sequence………………..……..…..………………38
  1. CLASS DIAGRAM
     1. Symbols of Class Daigrams…………….…….………….39
     2. Class Daigram…………………………….….…………..40

1. ENTITY RELATIONSHIP DIAGRAM
   1. Symbols of E R Diagram……………………………..….…....……43
   2. E R Daigram………………………………………….…..………...47
2. DATA DICTIONARY…………………………………..…….…...………..48
3. REFERENCES …………………………………….….…….……....………61

\

Project Profile

|  |  |
| --- | --- |
| **Project Title:** | **Mobile and Laptop Accessories Online Selling** |
| Internal Guide : |  |
| External Guide : |  |
| Project Duration : | 1 Year |
| Group Number : |  |
| Number of member : | 3 |
| Presented to : | Navgujarat Collage of Computer Applications,  Opposite Gujarat Vidhyapith,  Ashram road,Ahmedabad. |

Company Profile

|  |  |
| --- | --- |
| **Company Name** | **Vytech Enterprise** |
| **Company Address** | 510, Addor Aspire, B/s Jhanvee Restaurant, Nr. Passport Office, Polytechnic Rd, Ahmedabad, Gujarat 380015 road,ramol,Ahmedabad-382449. |
| **Contact Person** | Zeel Thakkar |
| **Contact No.** | 989888180 |

ANALAYSIS

EXISTING SYSTEM:-

* Buyer’s satisfaction is the first priority of the business .
* So We are providing an online platform to so that the customers can buy or see various types of Mobiles , Laptop Products.
* It helps buyers to Mobile and Laptop Products according to their requirements and needs.
* using Our website customers are knows about latest Mobile and Laptop Products and their offers online.
* buyers don’t need to go outside and search.
* Using our website sellers don’t need to tensed about their Mobile and Laptop marketing.
* The Main thing is a customer are easily purchase items by online.

### PROPOSED SYSTEM (NEW SYSTEM):-

* The Online Shopping Website Which can access anywhere and any time .
* A Buyer can Contact Directly and Order from our Website
* . Our Website Which Avoids the Difference between Buyer and seller .
* This Website keeps the data in a centralized way which is available to all the users simultaneously .
* It is very easy to manage historical data in database .
* No Specific training is required for the distributors to use this application .
* They can easily use the tool that decreases manual Hours spending for normal things and hence increases the performance .

TOOLS AND TECHNOLOGY:-

**1)TECHNOLOGY:-**

1. **Front End :**

Html , CSS & Bootstrap .

**2 . Backend :**

PHP , MySql .

**2) Tools:-**

* Xampp server
* Visual Studio Code
* Microsoft word
* Microsoft PowerPoint
* Draw.io
* Chrome

### **SYMBOLS OF SYSTEM FLOW DIAGRAM**

System flow diagram is a basically a graphical and sequential representation of the major steps involved in a systematic process.

**SYMBOLS OF SYSTEM FLOW DIAGRAM**

❖ **Start/end point**



An oval represents a start or end point.

The terminator symbol marks the starting or ending point of the system. It usually contains the word “Start” or “End”.

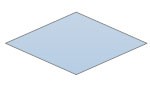
❖ **Arrows**

A line is a connector that shows relationships between the representative shapes.

❖ **Process**

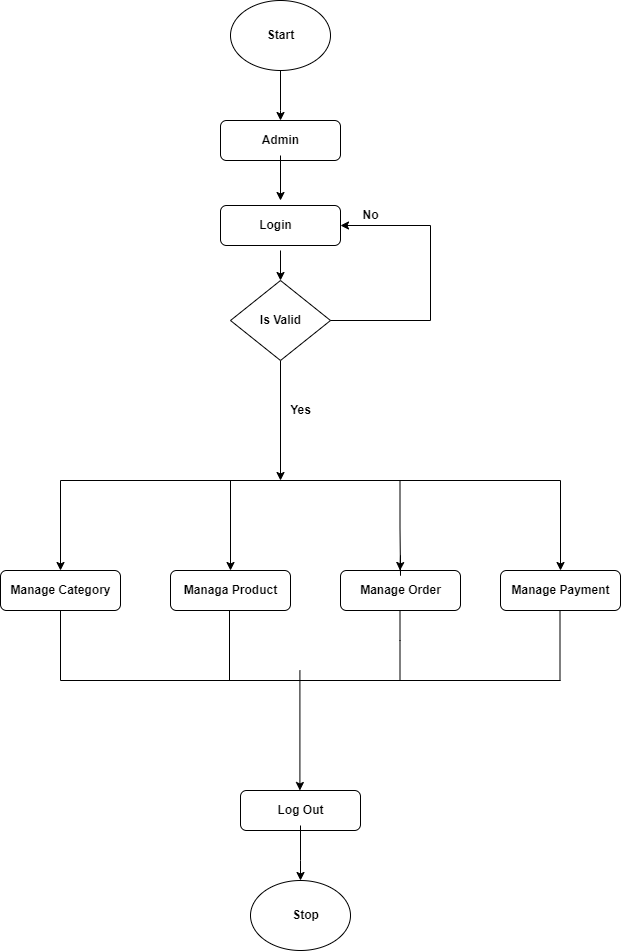


A rectangle represents a process. ❖ **Decision**

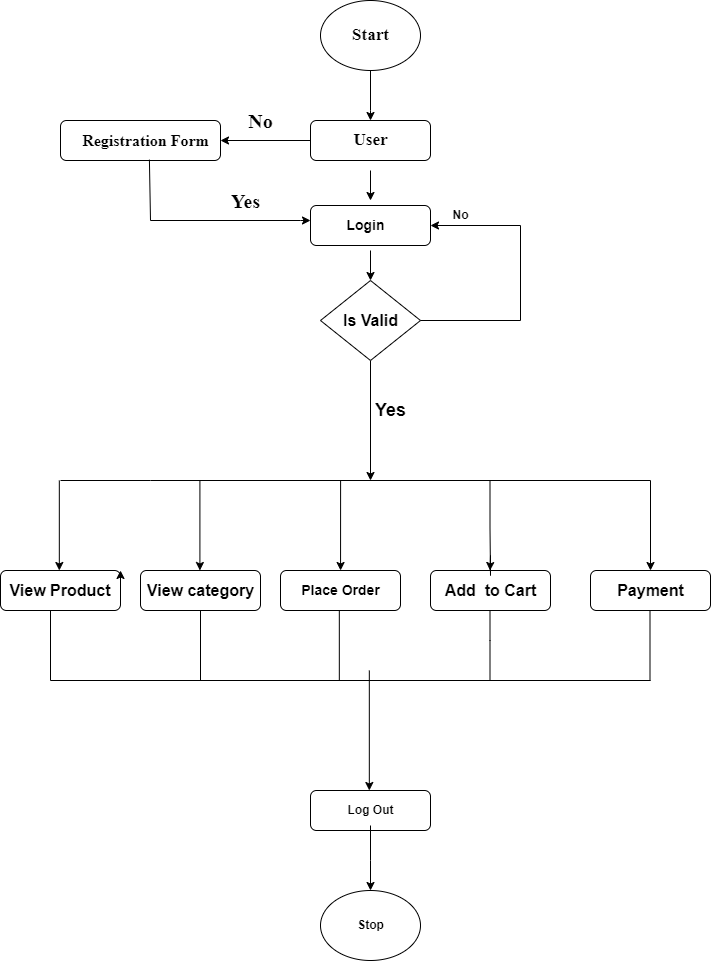


A diamond indicates a decision. It appoints where the outcome of a decision dictates the next step. There can be multiple outcomes, but often there are just two – yes and no or true – false, then branches to different parts of the systemflow diagram**.**

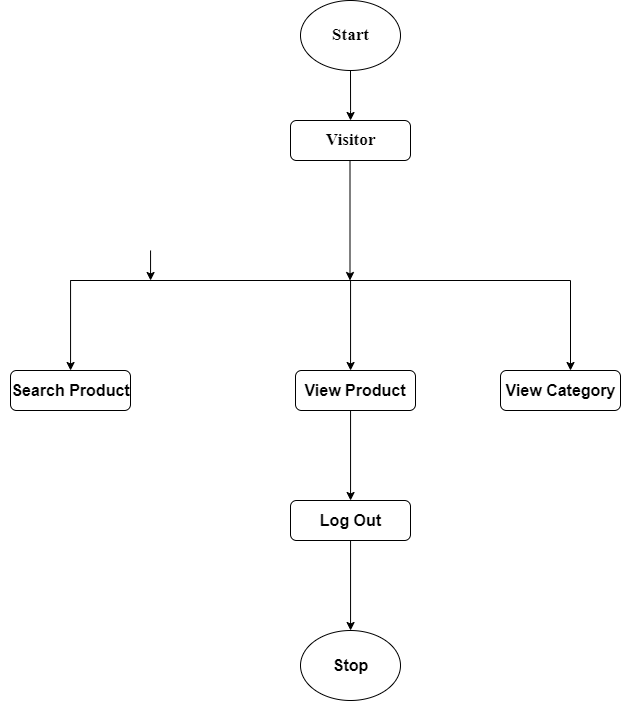
### **ADMIN SYSTEM FLOW DIAGRAM**



### **USER SYSTEM FLOW DIAGRAM**

****

### **VISITOR SYSTEM FLOW DIAGRAM**



## UML DAIGRAMS

### 5.1 USE CASE DAIGRAM

* Use case is a set of scenarios that describing an interaction between a user and a system.
* Use case diagram displays the relationship among actors and use cases.
* The two main components of a use case diagram are use cases and actors.
* An actor represents user or another system that will interact with the system.
* Use case is an external view of the system that represents some actions the user might perform in order to complete a task.

#### 5.1.1 SYMBOLS OF USE CASE DIAGRAM

##### **❖** Use Case

Use case

Draw use cases using ovals. Label the ovals with verbs that represent the system’s functions.

##### **❖** Actors



Actors are the users of a system. When one system is the actor of another system label the actor system with the actor stereotype.

##### **❖** System

|  |
| --- |
|  |

System

Draw your system’s boundaries using a rectangle that contains use cases, Place actorbnoutside the system’s boundaries.

##### **❖** Relationships

<<include>>

Relationship

<<extend>>

Illustrate relationships between an actor and a use case with a simple line.

For relationships among use cases, use arrows labeled either “uses” or

“extends.” A “uses” relationship indicates that one use case is needed by another in order to perform a task.

An “extends” relationship indicates alternative options under a certain use case.

### **ADMIN USECASE DIAGRAM**



**ACTIVITY DIAGRAM**

* Activity diagrams are graphical representation of step wise activities and actions with Support for choice, iteration and concurrency.
* Activity diagram can be used to describe the business and operational step – by – step Workflows of components in a system.
* Activity diagrams are typically used for business process modeling, for modeling the logic captured by a single usage scenario, or for modeling the detailed logic.
* Although UML activity diagrams could potentially model the internal logic of a complex operation it would be far better to simply rewrite the operation so that it is simple enough that you don’t require an activity diagram.

**SYMBOLS OF ACTIVITY DIAGRAM**

❖ **Initial State or Start Point**

A small filled circle followed by an arrow represents the initial action state or the start point for any activity diagram. For activity diagram using swim lanes, make sure the start point is placed in the top left corner of the first column.



❖ **Activity or Action State**

An action state represents the non-interruptible action of objects. You can draw an action state in Smart Draw using a rectangle with rounded corners.



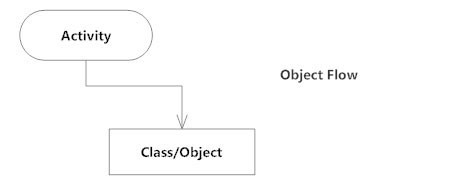
❖ **Action Flow**

Action flows, also called edges and paths, illustrate the transitions from one action state to another. They are usually drawn with an arrowed line.



❖ **Object Flow**

Object flow refers to the creation and modification of objects by activities. An object flow arrow from an action to an object means that the action creates or influences the object. An object flow arrow from an object to an action indicates that the action state uses the object.



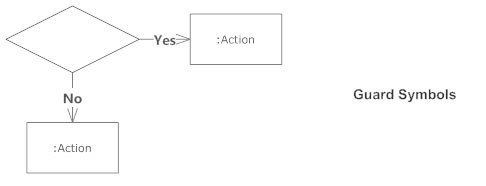
❖ **Decisions and Branching**

A diamond represents a decision with alternate paths. When an activity requires a decision prior to moving on to the next activity, add a diamond between the two activities. The outgoing alternates should be labeled with a condition or guard expression. You can also label one of the paths "else."



❖ **Guards**

In UML, guards are a statement written next to a decision diamond that must be true before moving next to the next activity. These are not essential, but are useful when a specific answer, such as "Yes, three labels are printed," is needed before moving forward.

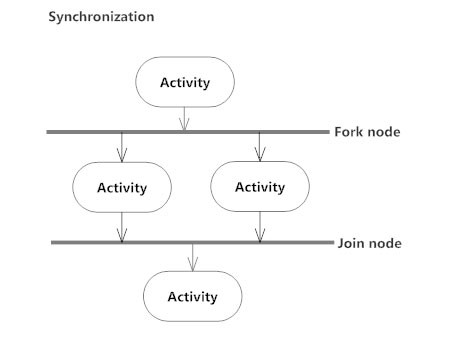


❖ **Synchronization**

A fork node is used to split a single incoming flow into multiple concurrent flows. It is represented as a straight, slightly thicker line in an activity diagram.

A join node joins multiple concurrent flows back into a single outgoing flow.

A fork and join mode used together are often referred to as synchronization.

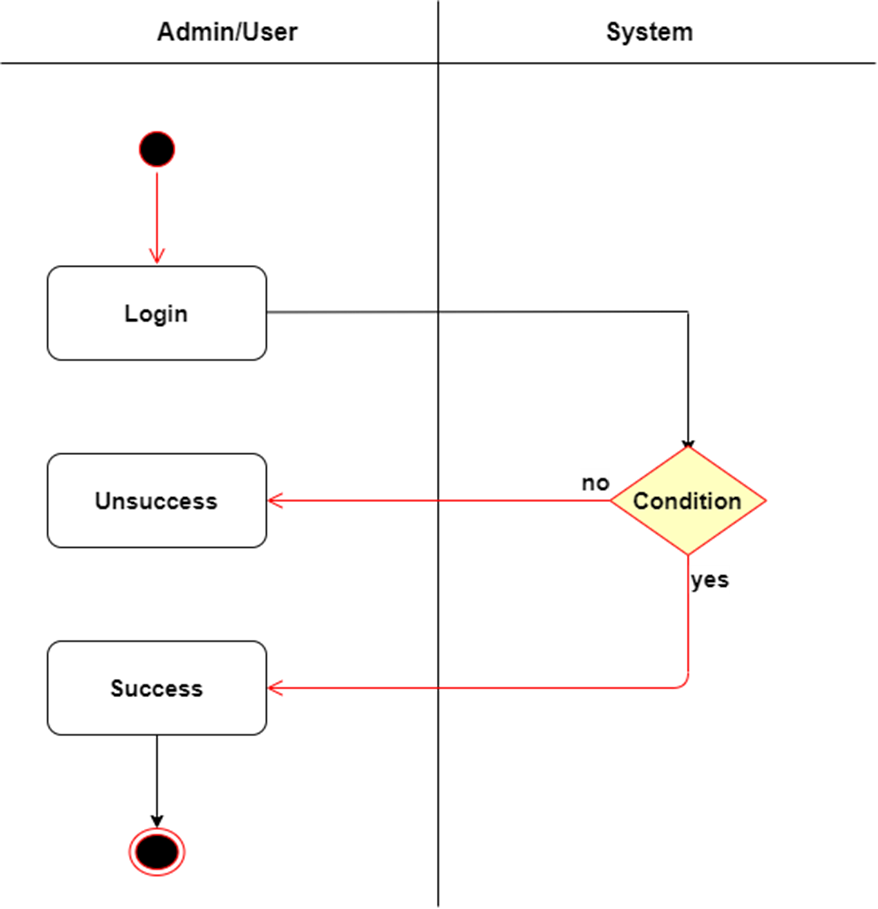


❖ **Final State or End Point**

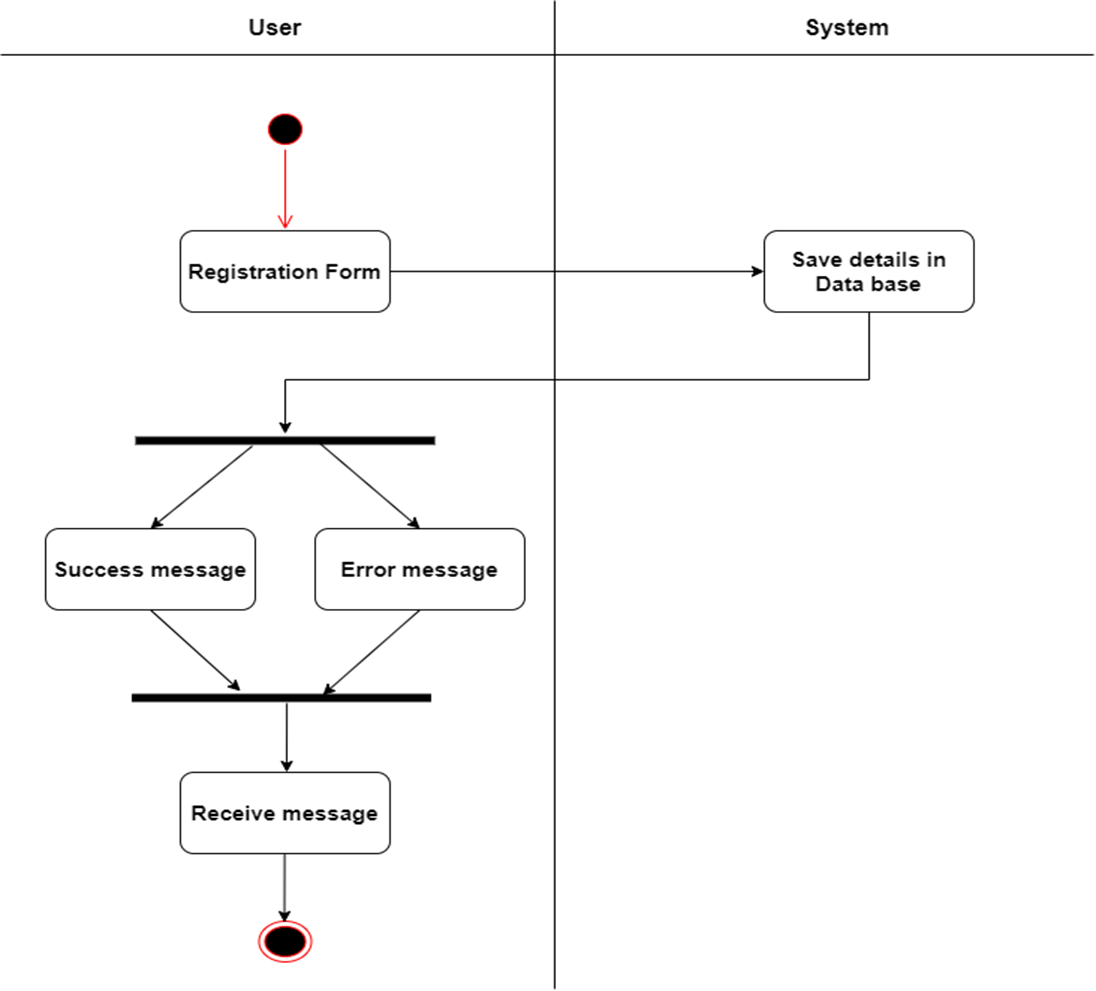
An arrow pointing to a filled circle nested inside another circle represents the final action state.



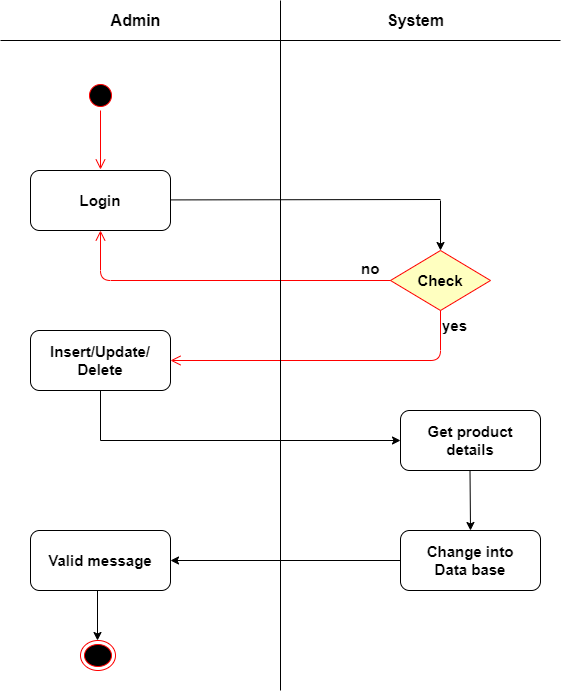
**ADMIN / USER LOGIN ACTIVITY**



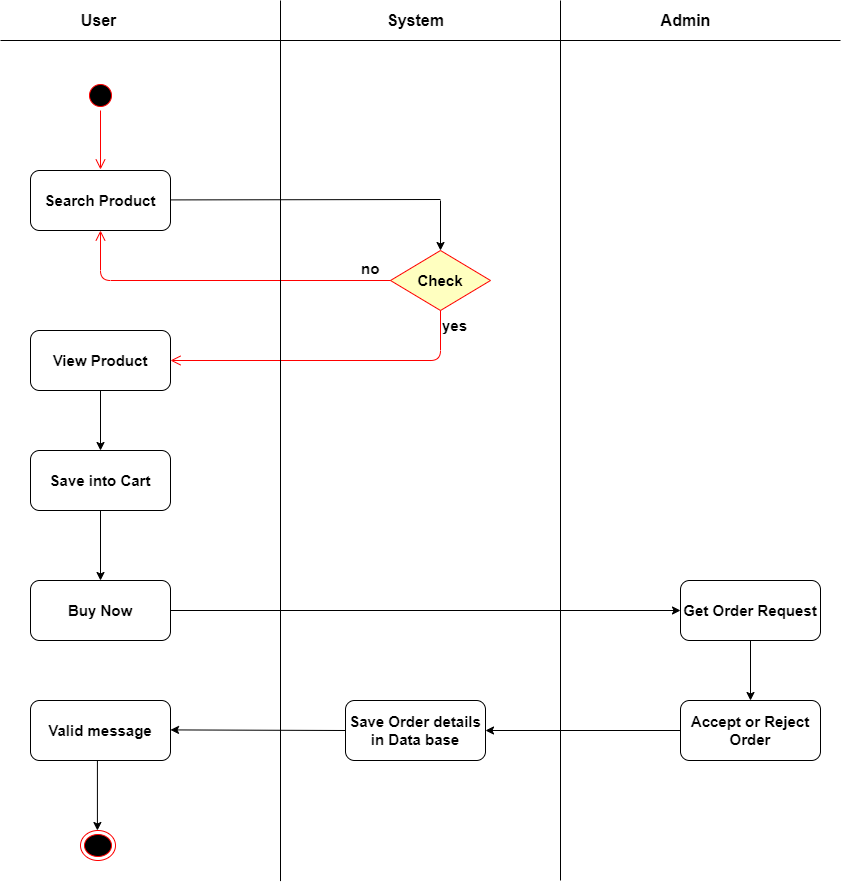
**REGISTRATION ACTIVITY**



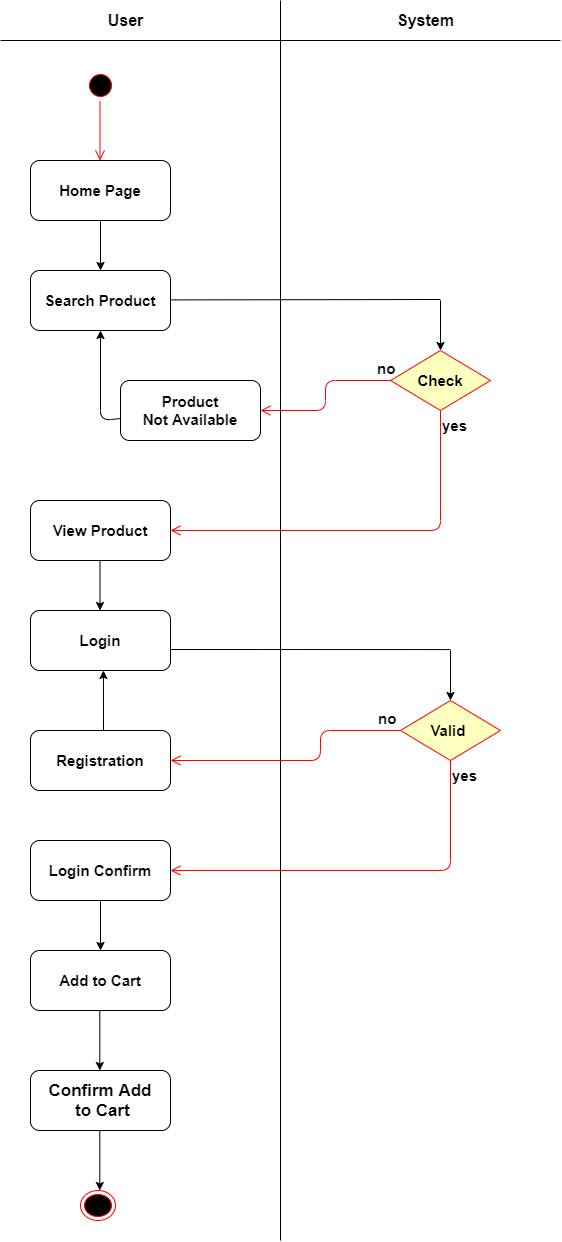
**INSERT PRODUCT ACTIVITY**



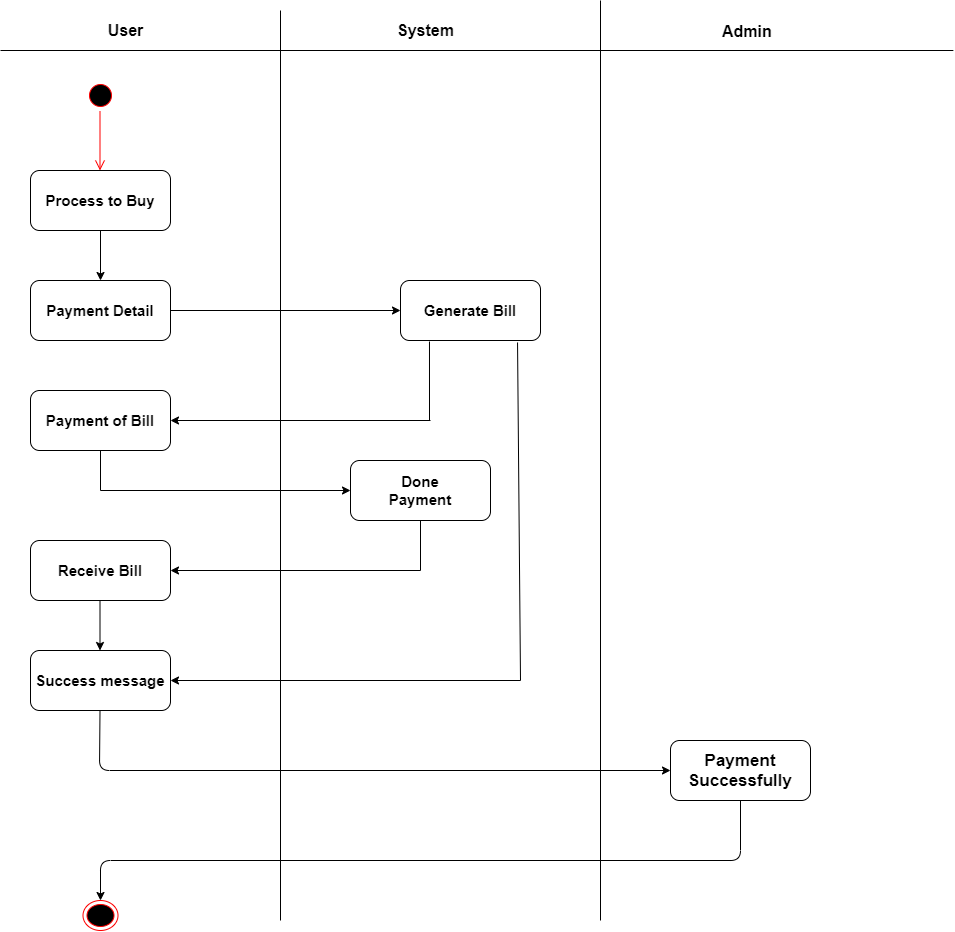
**ORDER ACTIVITY**



**CART ACTIVITY**



**PAYMENT ACTIVITY**



**SEQUENCE DIAGRAMS**

* A sequence diagram is a graphical view of a scenario that shows object interaction in a time-based sequence what happens first, what happens next.
* Sequence diagrams establish the roles of objects and help provide essential information to determine class responsibilities and interfaces.
* This type of diagram is best used during early analysis phases in design because they are simple and easy to comprehend. Sequence diagrams are normally associated with use cases.

**SYMBOLS OF SEQUENCE DIAGRAM**

❖ **Class roles and participants**



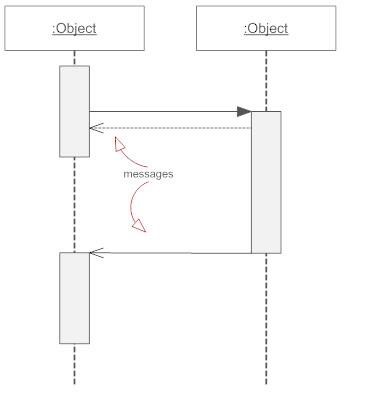
Class roles describe the way an object will behave in context. Use the UML objectsymbol to illustrate class roles, but don’t list object attributes.

❖ **Activation or Execution Occurrence**



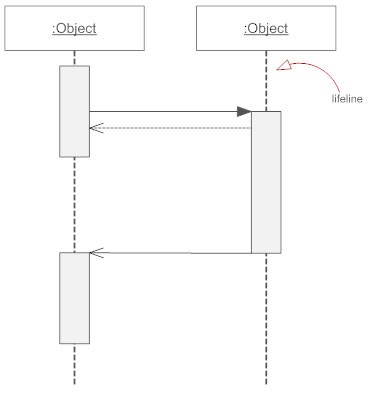
Activation boxes represent the time an object needs to complete a task. When an object is busy executing a process or waiting for a reply message, use a thin gray rectangle placed vertically on its lifeline.

❖ **Messages**



Messages are arrows that represent communication between objects. Use half-arrowed lines to represent asynchronous messages. Asynchronous messages are sent from an object that will not wait for a response from the receiver before continuing its tasks. For message types, see below.

❖ **Lifelines**



Lifelines are vertical dashed lines that indicate the object’s presence over time.

**Types of messages in sequence diagram**

❖ **Synchronous Message**



A synchronous message requires a response before the interaction can continue. It’s usually drawn using a line with a solid arrowhead pointing from one object to another.

❖ **Asynchronous Message**



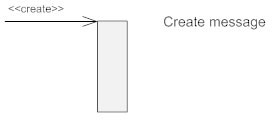
Asynchronous message doesn’t need a reply for interaction to continue. Like synchronous messages, they are drawn with an arrow connecting two lifelines; however, the arrowhead is usually open and there’s no return message depicted.

❖ **Reply or Return Message**



A reply message an object sends to itself, usually shown as a U-shaped arrow pointing back to itself.

❖ **Create Message**



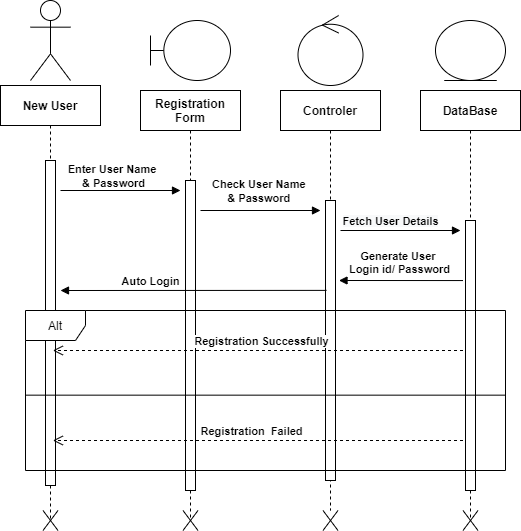
This is a message that creates a new object. Like a return message, it’s depicted with a dashed line and an open arrowhead that points to the rectangle representing the object created.

❖ **Delete Message**

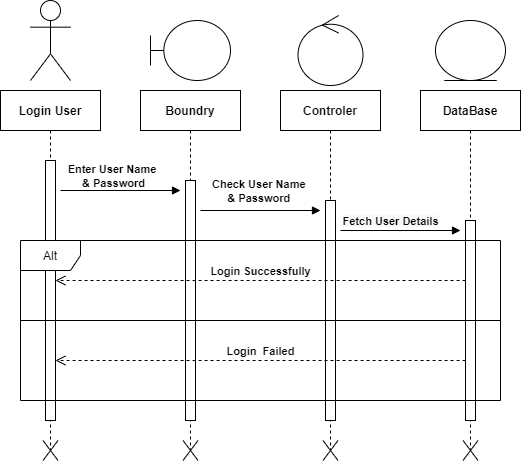


This is a message that destroys an object. It can be shown by an arrow with an x at the end.

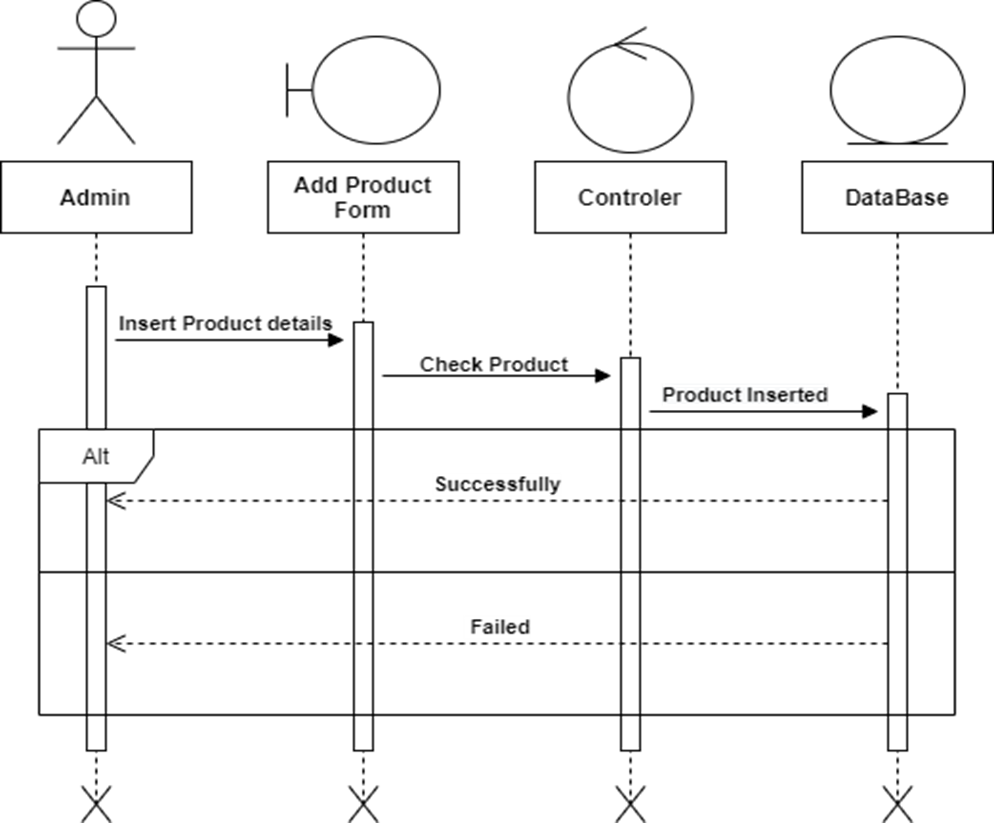
**REGISTRATION SEQUENCE**



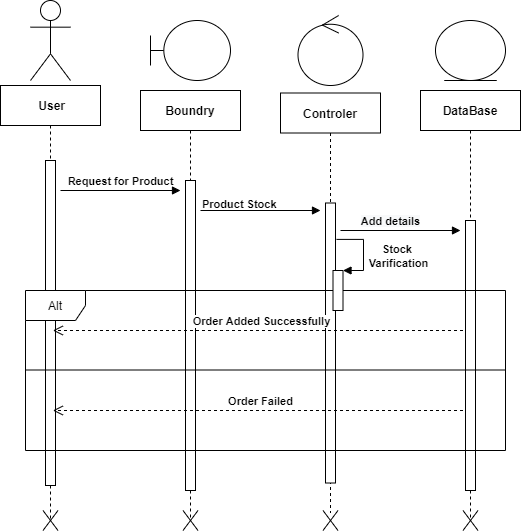
**LOGIN SEQUENCE**

****

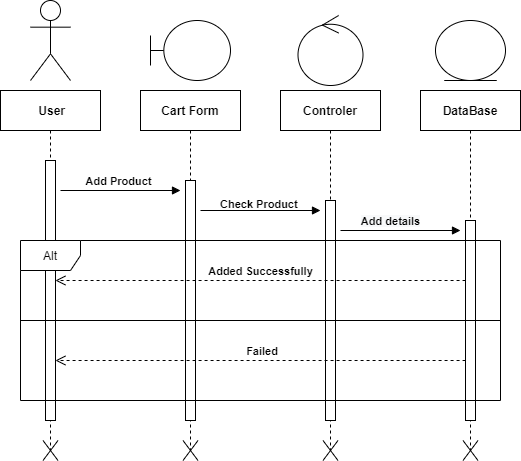
**INSERT PRODUCT SEQUENCE**



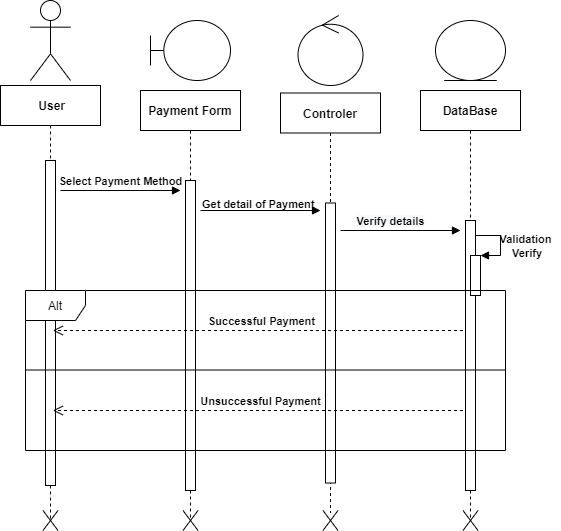
**ORDER SEQUENCE**



**CART SEQUENCE**



**PAYMENT SEQUENCE**



**CLASS DIAGRAM**

* A class diagram is a picture for describing generic descriptions of possible systems.
* Class diagrams and collaboration diagrams are alternate representations of object models.
* Class diagrams contain classes and object diagrams contain objects, but it is possible to mix classes and objects when dealing with various kinds of metadata, so the separation is not rigid we applied that concepts over here.
* Class diagrams contain icons representing classes, interfaces, and their relationships.

**SYMBOLS OF CLASS DIAGRAM**

❖ **Aggregation**



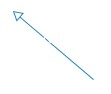
**Aggregation** is a special type of association in which objects are assembled or configured together to create a more complex object. An aggregation describes a group of objects and how you interact with them.

❖ **Dependency**



**Dependency** relationship is a relationship in which one element, the client, uses or depends on another element, the supplier.

❖ **Composition**



**Composition** represents whole-part relationships and is a form of aggregation.

❖ **Generalization**



**Generalization** is a relationship in which one model element (the child) is based on another model element (the parent).

❖ **Association**



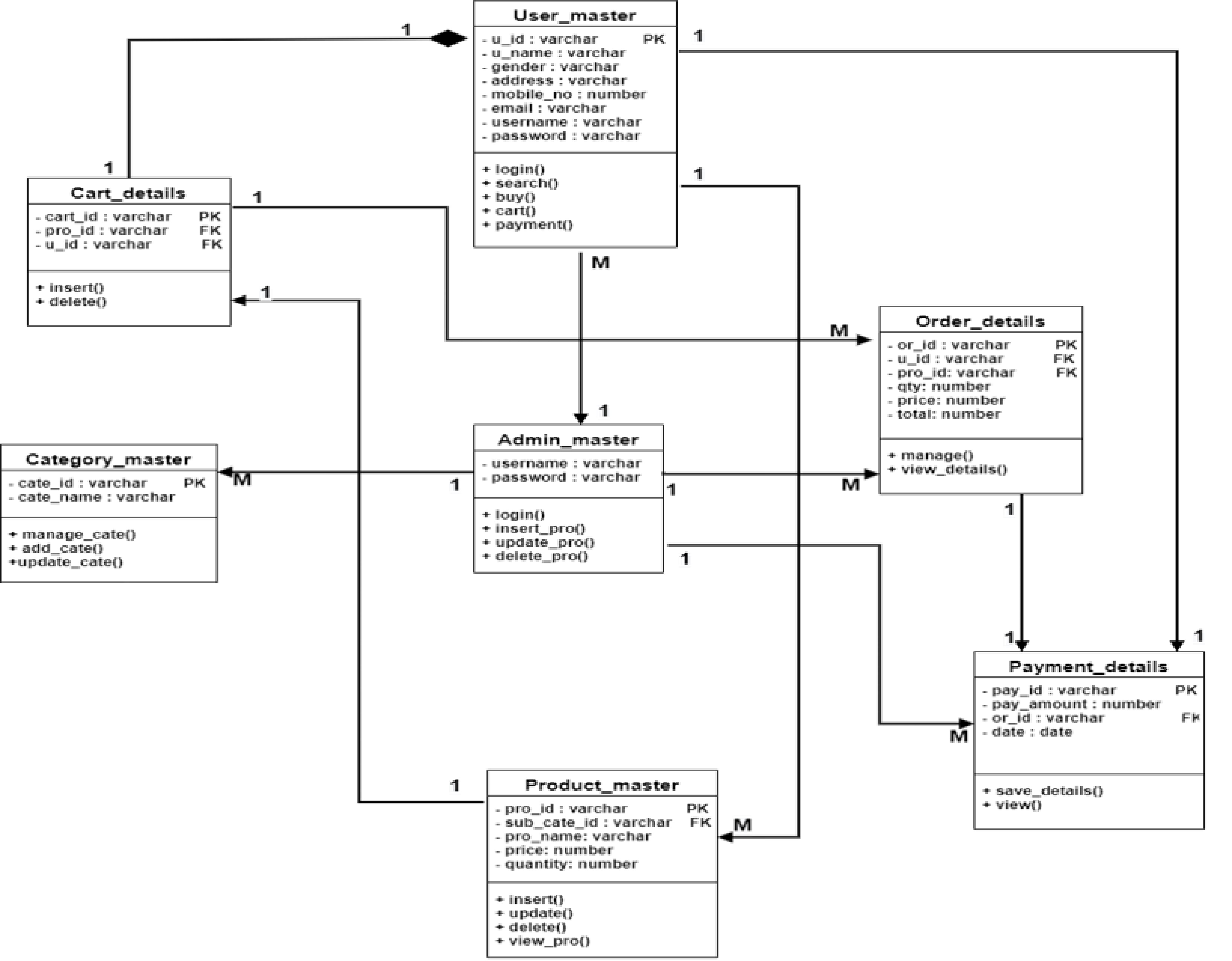
**Association** is a relationship between two classifiers, such as classes or use cases, that describes the reasons for the relationship and the rules that govern the relationship.

❖ **Constraint**

**Constraint** is an extension mechanism that enables you to refine the semantics of a UML

model element.

**CLASS DIAGRAM**



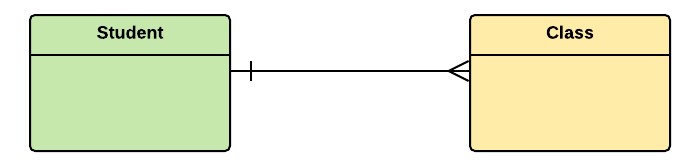
## E-R DIAGRAM

**ENTITY-RELATIONSHIP DIAGRAM (ERD)** displays the relationships of entity set stored in a database. In other words, we can say that ER diagrams help you to explain the logical structure of databases. At first look, an ER diagram looks very similar to the flowchart. However, ER Diagram includes many specialized symbols, and its meanings make this model unique. The purpose of ER Diagram is to represent the entity framework infrastructure.

### 6.1 SYMBOLS OF E R DIAGRAM

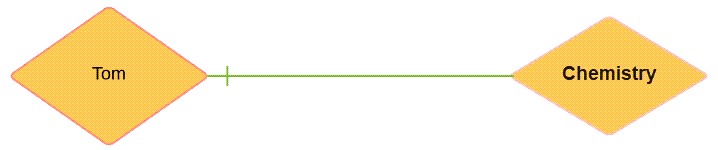
#### **❖** Entity set

An entity set is a group of similar kind of entities. It may contain entities with attribute sharing similar values. Entities are represented by their properties, which also called attributes. All attributes have their separate values. For example, a student entity may have a name, age, class, as attributes.



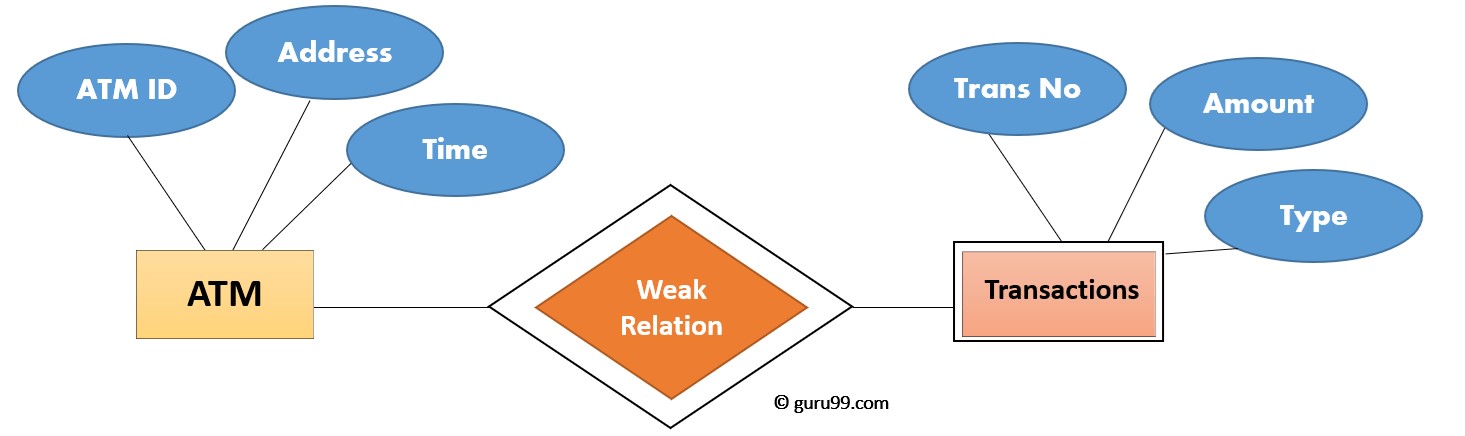
#### **❖** Relationship

Relationship is nothing but an association among two or more entities. E.g., Tom works in the Chemistry department.



#### **❖** Weak Entities

A weak entity is a type of entity which doesn't have its key attribute. It can be identified uniquely by considering the primary key of another entity. For that, weak entity sets need to have participation.

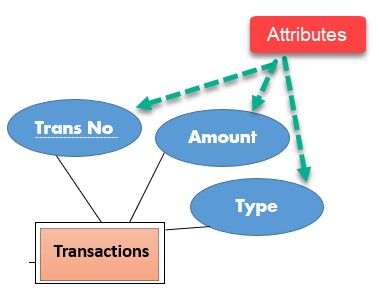


#### **❖** Attributes

It is a single-valued property of either an entity-type or a relationship-type.

For example, a lecture might have attributes: time, date, duration, place, etc.

An attribute is represented by an Ellipse

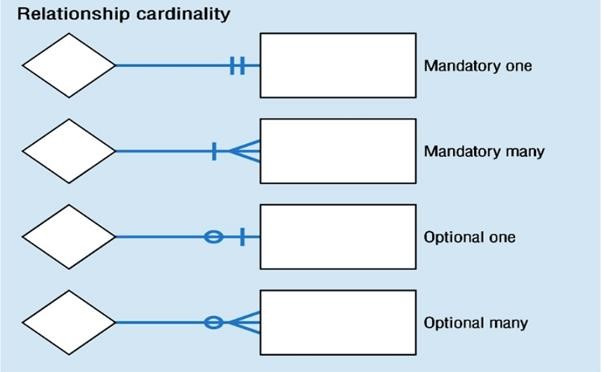


#### **❖** Cardinality

Defines the numerical attributes of the relationship between two entities or entity sets.

Different types of cardinal relationships are:

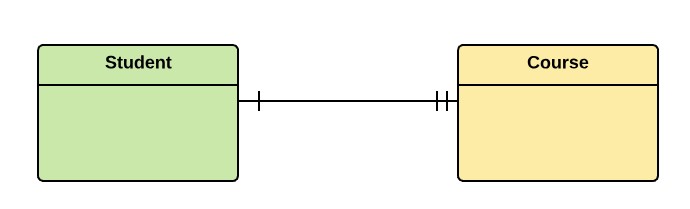
* One-to-One Relationships
* One-to-Many Relationships
* May to One Relationships
* Many-to-Many Relationships



**1.One-to-one:**

One entity from entity set X can be associated with at most one entity of entity set Y and vice versa.

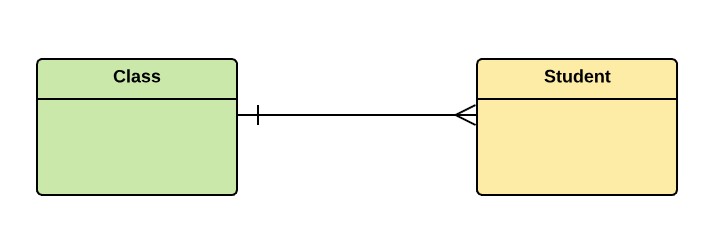
Example: One student can register for numerous courses. However, all those courses have a single line back to that one student.



**2.One-to-many:**

One entity from entity set X can be associated with multiple entities of entity set Y, but an entity from entity set Y can be associated with at least one entity.

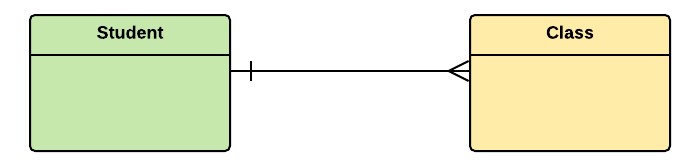
For example, one class is consisting of multiple students.



##### 3. Many to One

More than one entity from entity set X can be associated with at most one entity of entity set Y. However, an entity from entity set Y may or may not be associated with more than one entity from entity set X.

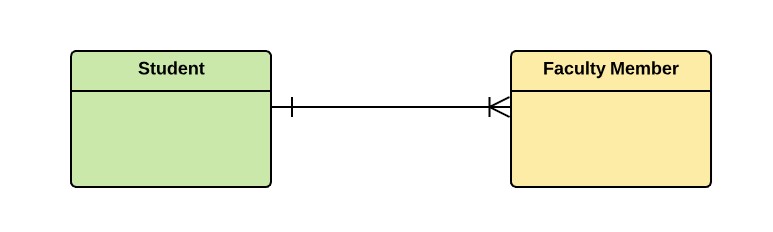
For example, many students belong to the same class.



**4. Many to Many:**

One entity from X can be associated with more than one entity from Y and vice versa.

For example, Students as a group are associated with multiple faculty members, and faculty members can be associated with multiple students.



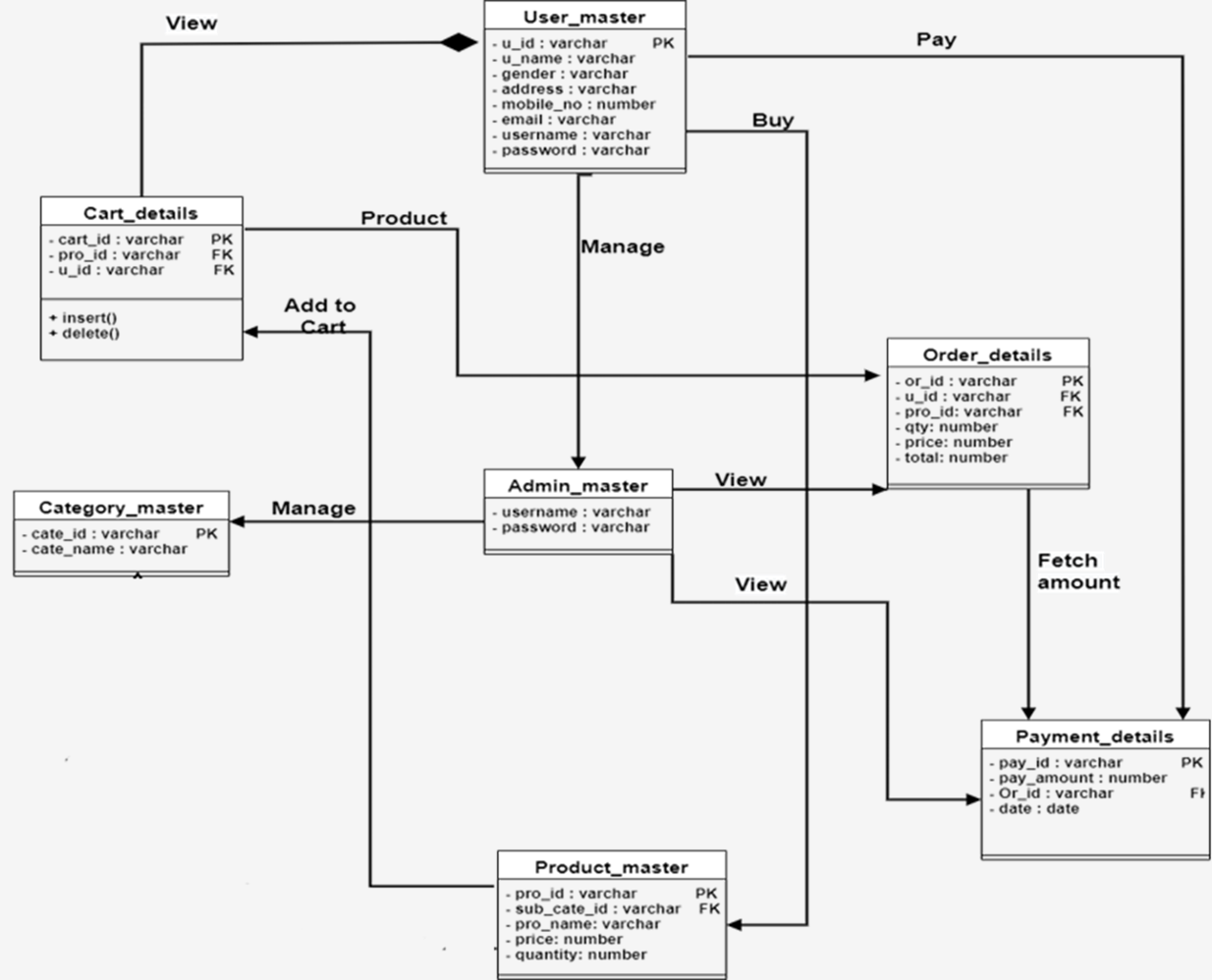
#### **❖** ER- Diagram Notations

ER- Diagram is a visual representation of data that describe how data is related to each other.

* **Rectangles:** This symbol represent entity types
* **Ellipses :** Symbolrepresent attributes
* **Diamonds:** This symbolrepresents relationship types
* **Lines:** It links attributes to entity types and entity types with other relationship types
* **Primary key:** attributes are underlined
* **Double Ellipses:** Represent multi-valued attributes



**E R DIAGRAM**

* 

****

## DATA DICTIONARY

* A data dictionary is a collection of descriptions of the data objects or items in a data model for the benefit of programmers and others who need to refer to them.

* A first step in analyzing a system of objects with which users interacts is to identify each object and its relationship to other objects. This process is called data modeling and results in a picture of object relationships.

* After each data object or item is given a descriptive name, its relationship is described (or it becomes part of some structure that implicitly describes relationship), the type of data (such as text or image or binary value) is described, possible predefined values are listed, and a brief textual description is provided. This collection can be organized for reference into a book called a datadictionary.

* When developing programs that use the data model, a data dictionary can be consulted to understand where a data item fits in the structure, what values it may contain, and basically what the data item means in real-worldterms.

### A DATA DICTIONARY CONTAINS

* The definitions of all schema objects in the database. (Tables, Views, Indexes, Clusters, Synonyms, Sequences, Procedures,

Functions, Packages, Triggers, and soon)

* How much space has been allocated for, and its currently used by, the schemaobjects.

* Integrity constraint information

.

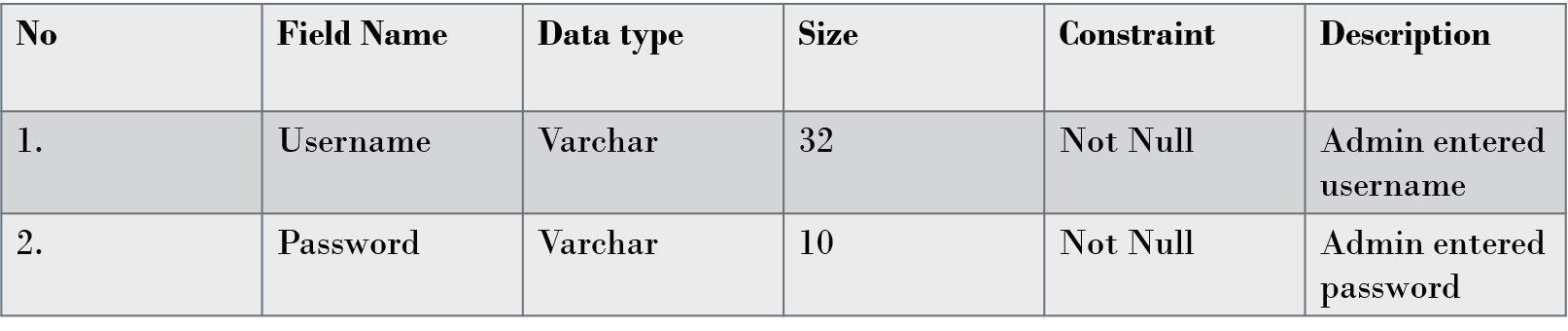
* The names of My SQL users.

* Privileges and roles each user has been granted.

* Auditing information, such as who accessed or updated various schema objects.

* Other general database information.

|  |  |
| --- | --- |
| **Table Name:** | **ADMIN** |
| **Table Description:** | This table contains details about ADMIN |
| **Constraint:** | Not Null |



|  |  |
| --- | --- |
| **Table Name:** | **USER** |
| **Table Description:** | **This table contains details about USER** |
| **Constraint:** | **Primary key, Not Null** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Data Type** | **Size** | **Constraint** | **Description** |
| **1** | **U\_id** | Varchar | **10** | **Primary Key** | **Auto increment** |
| **2** | **U\_name** | Varchar | **30** | **Not null** | **Userentered** |
| **3** | **Gender** | Varchar | **10** | **Not null** | **User select gender** |
| **4** | **Address** | Varchar | **50** | **Not null** | **Userentered** |
| **5** | **Mobile No** | Varchar | **10** | **Not null** | **Userentered** |
| **6** | **Email** | Varchar | **30** | **Not null** | **Userentered** |
| **7** | **Username** | Varchar | **15** | **Not null** | **Userentered** |
| **8** | **Password** | Varchar | **15** | **Not null** | **Userentered** |

|  |  |
| --- | --- |
| **Table Name:** | **PRODUCT CATEGORY** |
| **Table Description:** | **This table contains details about Category** |
| **Constraint:** | **Primary key, Not Null** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Data Type** | **Size** | **Constraint** | **description** |
| **1** | **Cate\_id** | **Int** | **3** | **Primary key** | **Auto increment** |
| **2** | **Cate\_name** | **varchar** | **30** | **Not null** | **Admin Entered** |

|  |  |
| --- | --- |
| **Table Name:** | **PRODUCT** |
| **Table Description:** | **This table contains details about Product** |
| **Constraint:** | **Primary key, Foreign Key , Not Null** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Data Type** | **Size** | **Constraint** | **Description** |
| **1** | **Pro\_id** | Varchar | **10** | **Primary Key** | **Auto increment** |
| **2** | **Pro\_name** | Varchar | **10** | **Not null** | **Admin entered pro\_name** |
| **3** | **Price** | Number | **15** | **Not null** | **Admin entered price** |
| **4** | **Quantity** | Number | **15** | **Not null** | **Admin entered quantity** |

|  |  |
| --- | --- |
| **Table Name:** | **ORDER** |
| **Table Description:** | **This table contains details about ORDER** |
| **Constraint:** | **Primary key, Foreign Key , Not Null** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Data Type** | **Size** | **Constraint** | **Description** |
| **1** | **Or\_id** | int | **10** | **Primary Key** | **Auto increment** |
| **2** | **U\_id** | int | **10** | **Foreign Key** | **From User table** |
| **3** | **Pro\_id** | int | **3** | **Foreign Key** | **From Product table** |
| **4** | **qty** | Number | **10** | **Not null** | **Product quantity** |
| **5** | **price** | Number | **10** | **Not null** | **Product Price** |
| **6** | **total** | Number | **10** | **Not null** | **Grand Total of the order** |

|  |  |
| --- | --- |
| **Table Name:** | **CART** |
| **Table Description:** | **This table contains details about CART** |
| **Constraint:** | **Primary key, Foreign Key , Not Null** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Data Type** | **Size** | **Constraint** | **Description** |
| **1** | **Cart\_id** | int | **3** | **Primary Key** | **Auto increment** |
| **2** | **Pro\_id** | int | **3** | **Foreign Key** | **From Product table** |
| **3** | **U\_id** | int | **3** | **Foreign Key** | **From User table** |

|  |  |
| --- | --- |
| **Table Name:** | **PAYMENT** |
| **Table Description:** | **This table contains details about PAYMENT** |
| **Constraint:** | **Primary key, Foreign Key , Not Null** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Data Type** | **Size** | **Constraint** | **Description** |
| **1** | **Pay\_id** | int | **10** | **Primary Key** | **Auto increment** |
| **2** | **Pay\_amount** | int | **10** | **Not null** | **Save Payment Amount** |
| **3** | **U\_id** | int | **10** | **Foreign Key** | **From Order details table** |
| **4** | **date** | Date | **10** | **Not Null** | **Save payment date** |

## REFERENCES

### **•** BOOKS

* System analysis and designing By Elias M. Awad
* Software Engineering By Roser S. Pearson
* Object Oriented Analysis and design By Grady Booch

### **•** WEBSITES

➢ W3Schools.com

THANK YOU

