# ACKNOWLEDGEMENT

I have a great pleasure for representing this project report entitle “Online KitchenWare” and I grab this opportunity to convey my immense regards towards all the distinguish people who have their valuable contribution in the hour of need.

I take this opportunity to thank “**Dr.Mrs. Usha Mukundan** ”, our principal of “**Ramniranjan Jhunjhunwala College**” , Ghatkopar(W) for giving me an opportunity to study in the institute and the most needed guidance throughout the duration of the course.

I also like to extend my gratitude to “**Mrs. Vaidehi Deshpande**”, Head of the Department of Computer Science for their timely and prestigious guidance.

I would also like to thank “ **Mrs. Nilam Patade** ” providing the guidance and necessary support during each phase of the project.

I also owe to my fellow friends who have been a constant source of help to solve the problem and also help during the project development process.

THANK YOU

DEEPA YADAV

# INTRODUCTION

## SCOPE OF THE SYSTEM

The customers Register with their details and get authentication for an authorized Login.

The software provides the following facilities to the customers:

Facilities easy shopping online anywhere with free shipping.

Provides information about the products in categories.

Customers are provided with up to date information on the products available.

The objective of this software is to provide easy assistance to both the customer as well as the merchant with proper database and information.

# EXIXTING SYSTEM

- In the existing system all transactions, dealings of products, purchasing of product are done manually which is time consuming.
- Reports are prepared manually as and when needed.
- Maintaning of reports is very tedious tasks.
- To buy any product user has to collect information about it either by visiting the shop or by asking people which is the better one.

# LIMITATIONS OF EXISTING SYSTEM

- Delay in delivery.
- Lack of significant discount in online shopping.
- Lack of shopping experience.
- Frauds in online shopping.

- The existing system faces many problems in maintaining large amount of data and retrieval of the data. These problems are,

## MANUAL ERRORS:-

- The organization presently holds a manual system which is unable to cope up with increasing structure of the products.
- Since the records are maintained manually, searching for a particular record and maintaining all record is difficult.
- All records and details are maintained in register books.

- Particular record and maintaining all record is difficult.
- All records and details are maintained in register books.
- There is a large chance of Human error in calculations maintenance.

## TIME CONSUMING:-

- A lot of paper work has to be done which is very time consuming.

- Searching records, details from register is very tedious job due to excess paper work.

**RISKY:-**

- There are no provisions for security.
- Due to this company's confidential data at risk.

## TEDIOUS JOBS:-

- In the current system, there is a chance of data redundancy therefore maintenance and updating is difficult.

# PROPOSED SYSTEM

## Features :-

- All the records and data have been computerized and are now persistent.
- Manual calculations have been automated.
- Access to all types of records made easy.
- This system will reduce the complexity to store all the information manually.
- It will also reduce the confusion created due to the improper manual storage of the information.

## Advantages of Proposed System:-

- This system is user-friendly, interactive and easy to use.
- Save excess paper work.
- Time saving.
- No complexity.
- Data security and backup facility.
- Easy navigation through the records is provided.
- Formatted data.
- Records/information's are easily approachable.
- 24x7 availability.
- Creates and manage new users easily.
- Provides security to data through authorization.

# FEASIBILITY STUDY

- Preliminary Investigation is the first phase in any system developing life cycle. The Feasibility Study is a major part of this phase.
- Feasibility Study means selecting the best system that meet the performance requirement.
- It is the measure of how beneficial or practical the development of an Information System would to the organization.
- Our study of the feasible development of the software is going to be in terms of the following aspects.

## Time Feasibility:-

As the registration, login options is given online so it will save the time of the member.

## Operational Feasibility:-

Proposed system is beneficial only if it can be turned into system that will meet the need of the client's operating requirements. The proposed system is operationally feasible due to the following reasons:

- It is easy to use and is very simple.
- The proposed Event Management website will cost no harm to the pre-school, instead it will enhance the result in a better respect and change of data can be done easily.
- The new website will avoid confusion and resistance by catching the user's attention as it is presentable.

## Economical Feasibility:-

1. Development Cost.
2. At first it is necessary to check that the proposed system is technically feasible to carry out the project. If they are not available, then find out the solution to obtain them.
   - Equipment's maintenance is also minimum.
   - Saving of paper work and manpower reduced.
   - Once the required hardware and software requirements get fulfilled there is no need for the user of our system to spend for any additional overhead.
   - Benefits which cannot be measured.
   - Increased Users Loyalty.
   - Increased Users satisfaction.

## Social Feasibility:-

Although generally there is always resistance, initially to any change in the system is aimed at relieving the workload of the users to extent the system is going to facilitate user to perform online get information about Quiz, Login etc. Thus there is no reason to make system socially unfeasible.

## TECHNICAL REQUIREMENTS

### Software:-

- Operating System: Windows 10
- Graphics: Screen Resolution 1024*768
- Color Quality: Highest 32 bits
- Frontend: Microsoft Visual Studio 2010
- Backend: SQL Management Studio

### Hardware:-

- Intel Pentium CPU N3710 @1.60Hz
- RAM-4.00GB.
- 64 bit operating system,x64-bassed processor
- Keyboard
- Mouse

# GANTT CHART

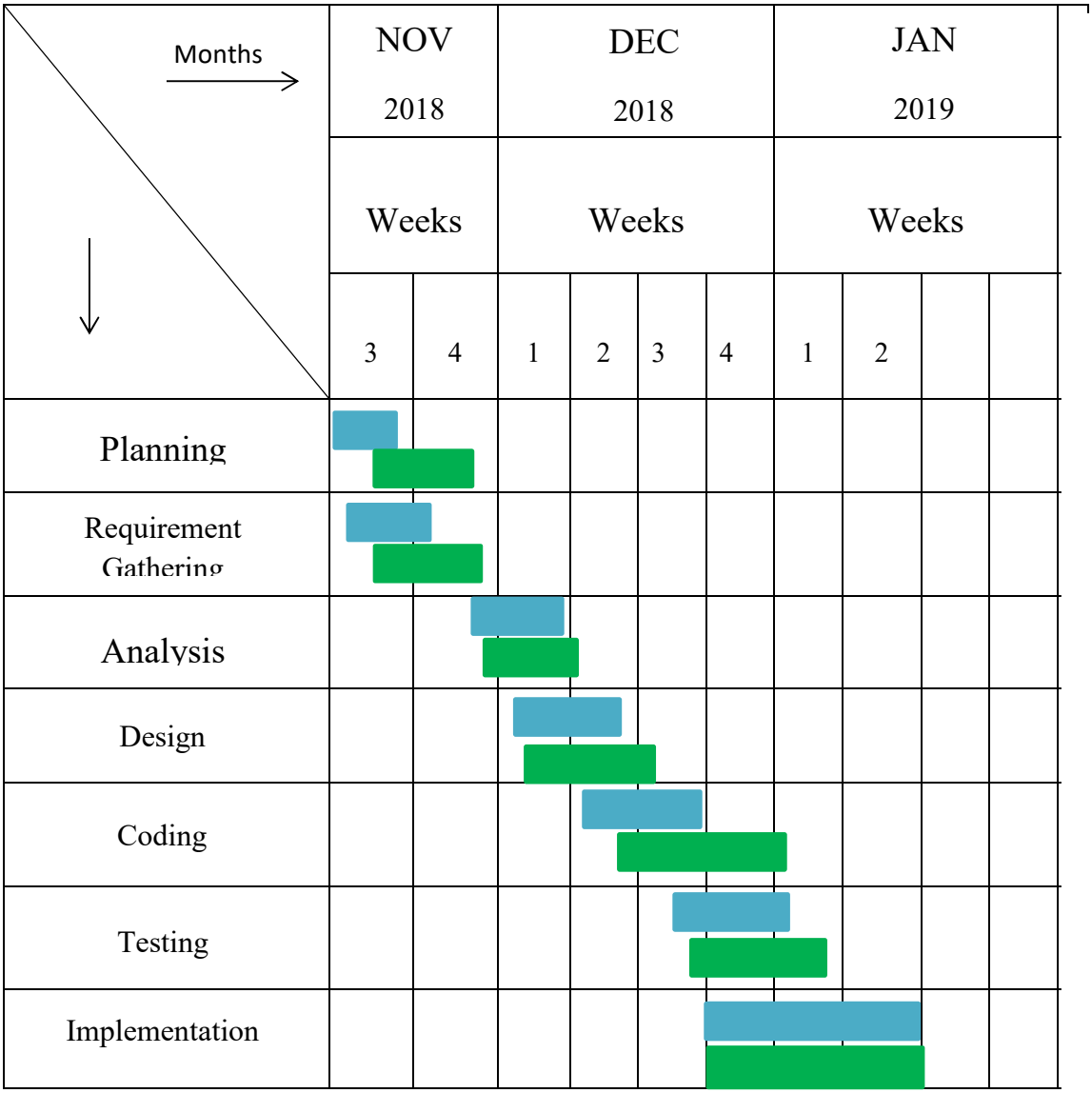
A Gantt chart is a type of bar chart that illustrates a project schedule. A Gantt Charts illustrates the start and finish dates of the terminal elements and summary elements of a project. Terminal elements and summary elements comprise the work breakdown structure of the project.

Gantt chart (Gantt Charts) are useful tools for analysing and planning more complex projects.

They:

- Help you to plan out the tasks that need to be completed.
- Give you a basis for scheduling when these tasks will be carried out.
- Help you to work out critical path for a project where you must complete it by a particular date.

When a project is under way. Gantt chart helps you to monitor whether the projects is on schedule. If it is not, it allows you to pinpoint the remedial action necessary to put it back on schedule. This chart is also used in Information Technology to represent data that has been collected.




Planning:

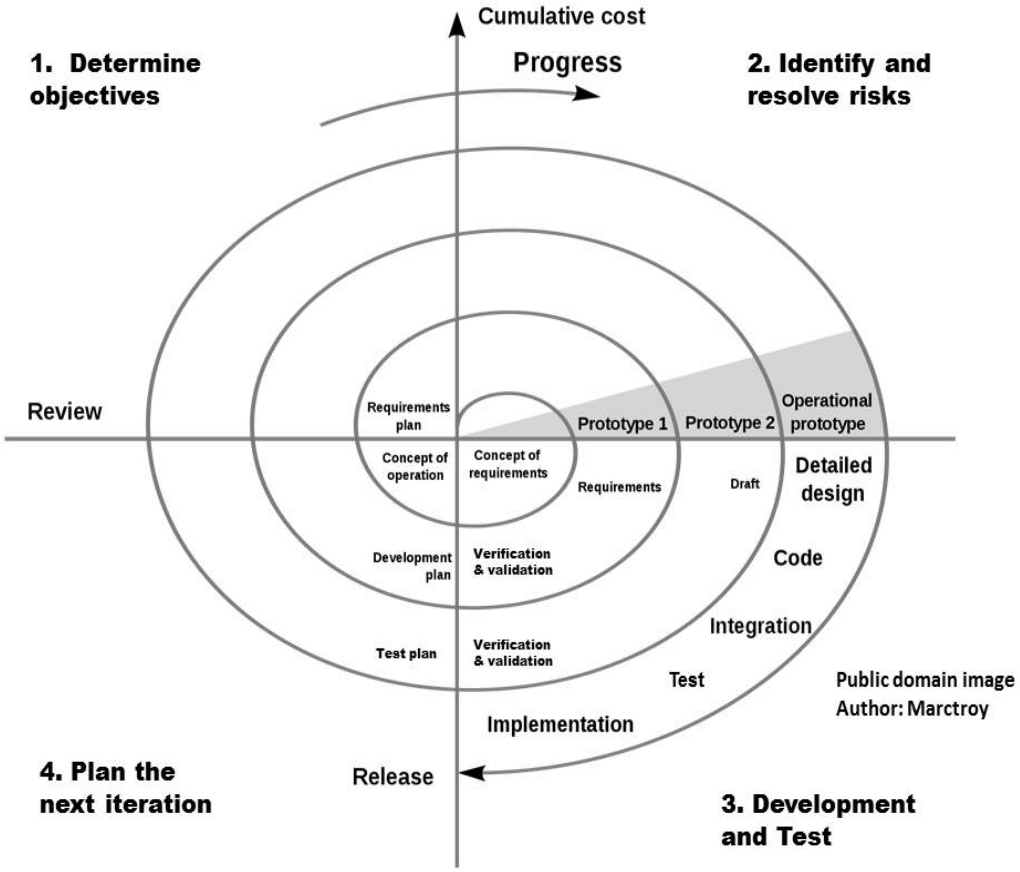
Execution:

# SYSTEM DESIGN

## SPIRAL MODEL:

**Spiral model** is one of the most important Software Development Life Cycle models, which provides support for **Risk Handling**. In its diagrammatic representation, it looks like a spiral with many loops. The exact number of loops of the spiral is unknown and can vary from project to project. **Each loop of the spiral is called a Phase of the software development process.** The exact number of phases needed to develop the product can be varied by the project manager depending upon the project risks. As the project manager dynamically determines the number of phases, so the project manager has an important role to develop a product using spiral model.
The Radius of the spiral at any point represents the expenses(cost) of the project so far, and the angular dimension represents the progress made so far in the current phase.

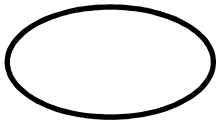
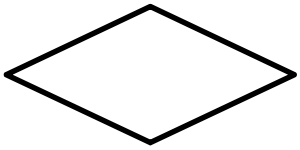
Below diagram shows the different phases of the Spiral Model:



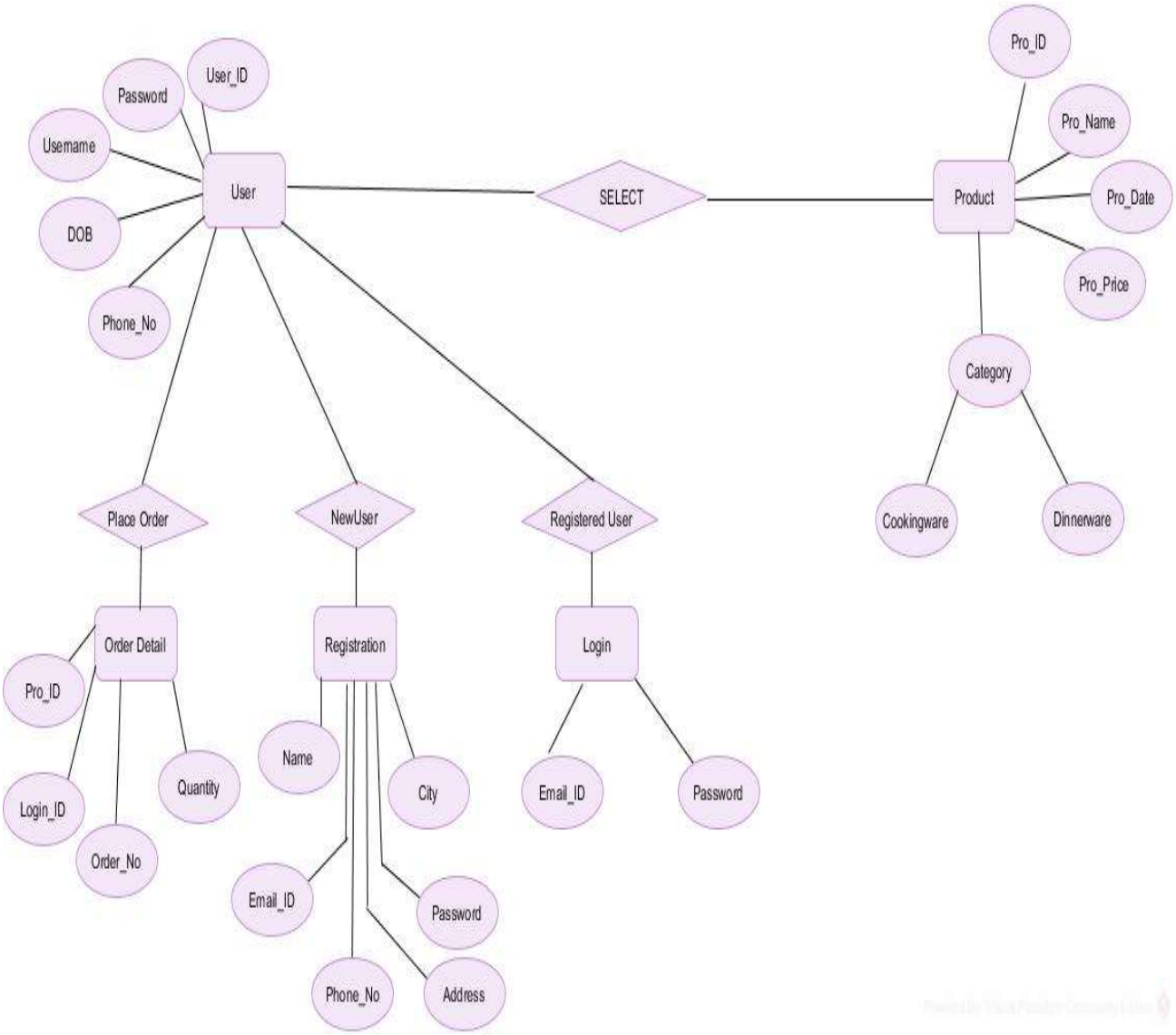
# UML DIAGRAMS

## ENTITY-RELATIONSHIP DAIGRAM

- This document is an entity-relationship diagram, or "ERD", for a system to manage Inventory Management System.
- An **entity–relationship model** describes interrelated things of interest in a specific domain of knowledge.
- A basic ER model is composed of entity types (which classify the things of interest) and specifies relationships that can exist between instances of those entity types.
- ERD is usually drawn in a graphical form as boxes (*entities*) that are connected by lines (*relationships*) which express the associations and dependencies between entities.
- An ER model is typically implemented as a database. In a simple relational database implementation, each row of a table represents one instance of an entity type, and each field in a table represents an attribute type.
- This ERD is of the latter type, intended to present an abstract, theoretical view of the major entities and relationships needed for management of electronic resources.
- It may assist the database design process for an e-resources management system, bust does not identify every table that would be necessary for an electronic resource management database.

| SYMBOLS | SHAPE NAME | DESCRIPTION |
|---|---|---|
|  | Entity | An entity is represented by a rectangle which contains the entity's name. |
|  | Attribute | Each attribute is represented by an oval containing attribute's name in diagram. |
|  | Relationship | Actions, which are represented by diamond shapes, show how two entities share information in the database. |
|  | Line | Solid lines that connect attributes to show relationships of entities. |

# ENTITY-RELATIONSHIP DAIGRAM

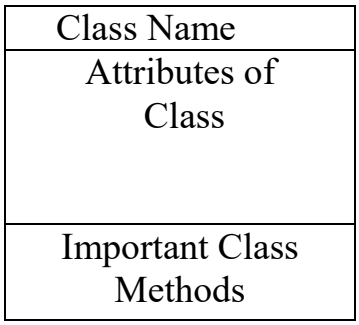


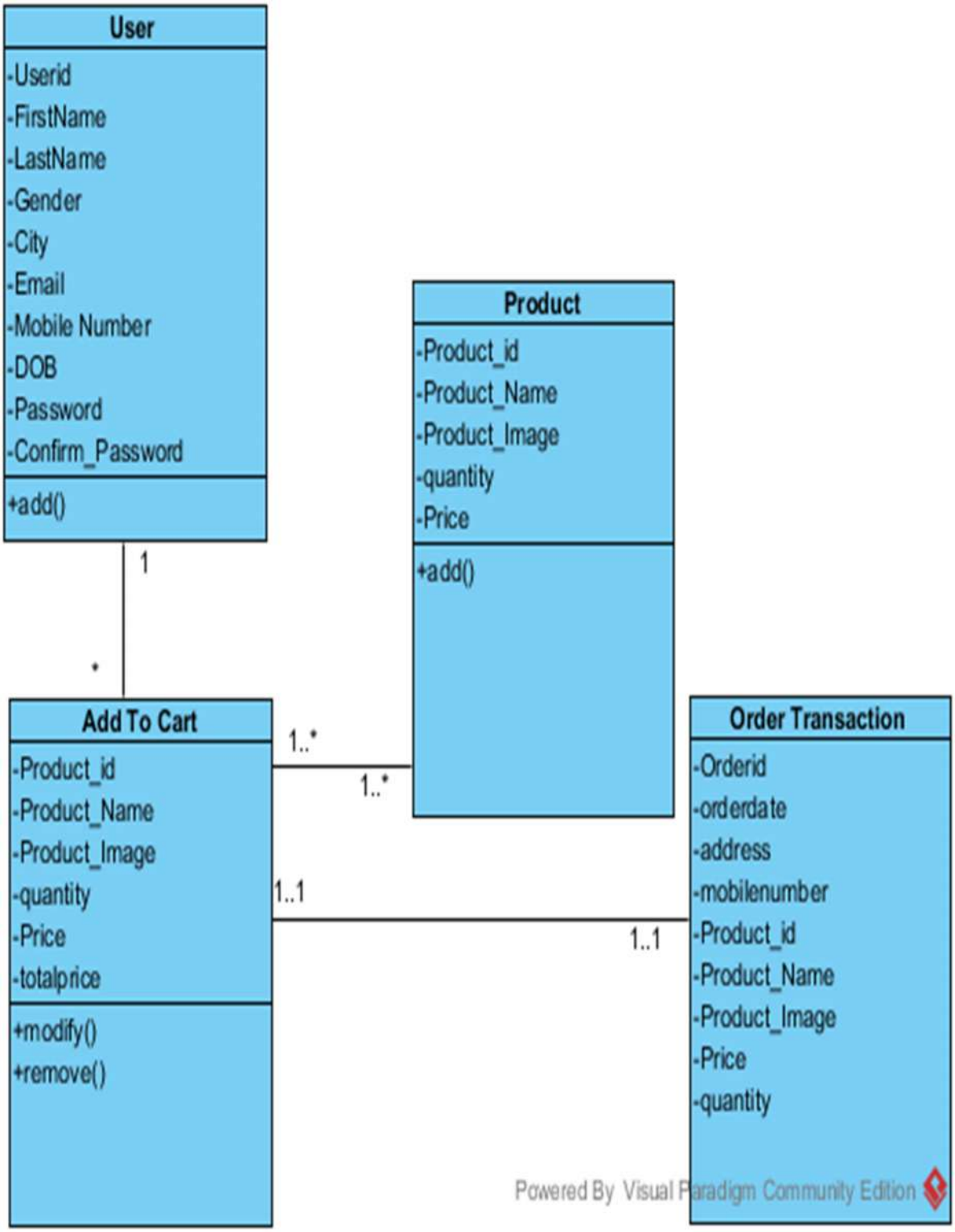
# CLASS DIAGRAM

It is model which is used to show the classes constituting a system and their interrelationship. It is based on UML. Only the important attributes and methods are shown in Class diagrams. In the initial period of analysis, the important attributes of the classes, which must be captured and the functionalities provided by the class may not be very clear. As the analysis progresses, the attributes and methods may be added. If more focus is on interrelationships of classes, then the attributes and methods may not be shown in the class diagram.

The class diagram is used to identify and classify the objects which constitute a system. It also includes the important attributes of the objects which must be captured.

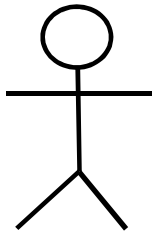
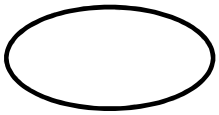
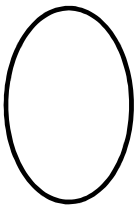
## NOTATION:-



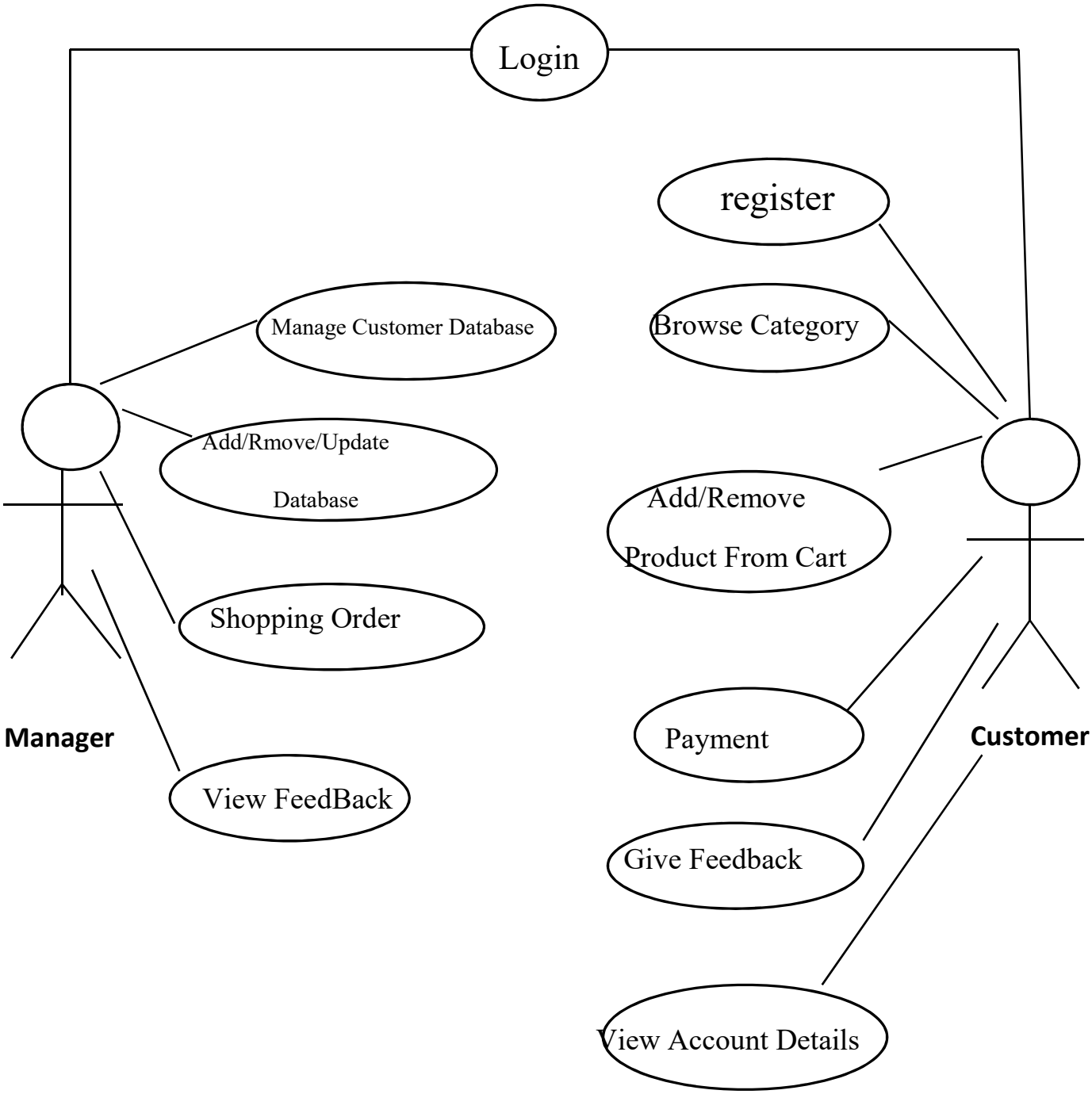

User
-Userid
-FirstName
-LastName
-Gender
-City
-Email
-Mobile Number
-DOB
-Password
-Confirm_Password
+add()
1
*
Product
-Product_id
-Product_Name
-Product_Image
-quantity
-Price
+add()
Add To Cart
-Product_id
-Product_Name
-Product_Image
-quantity
-Price
-totalprice
+modify()
+remove()
1..*
1..*
1..1
1..1
Order Transaction
-Orderid
-orderdate
-address
-mobilenumber
-Product_id
-Product_Name
-Product_Image
-Price
-quantity
Powered By Visual Paradigm Community Edition

# USE CASE DIAGRAM

A use case diagram in the unified modelling language is a type of behavioural diagram defined by and created from a Use-Case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

| SYMBOL | SHAPE NAME | DESCRIPTION |
|---|---|---|
|  | Actor | In UML, the person involved is called an Actor. An Actor is always outside of the Automation Boundary of the System. |
|  | Use Case | It describes an activity the system carries out in response to an event. |
|  | Connecting line | The arrow is used to show which Actors participate in which use cases. |
|  | Automation Boundary | It is Boundary between Use Case and Actor. |

# USE CASE DIAGRAM



# ACTIVITY DIAGRAM

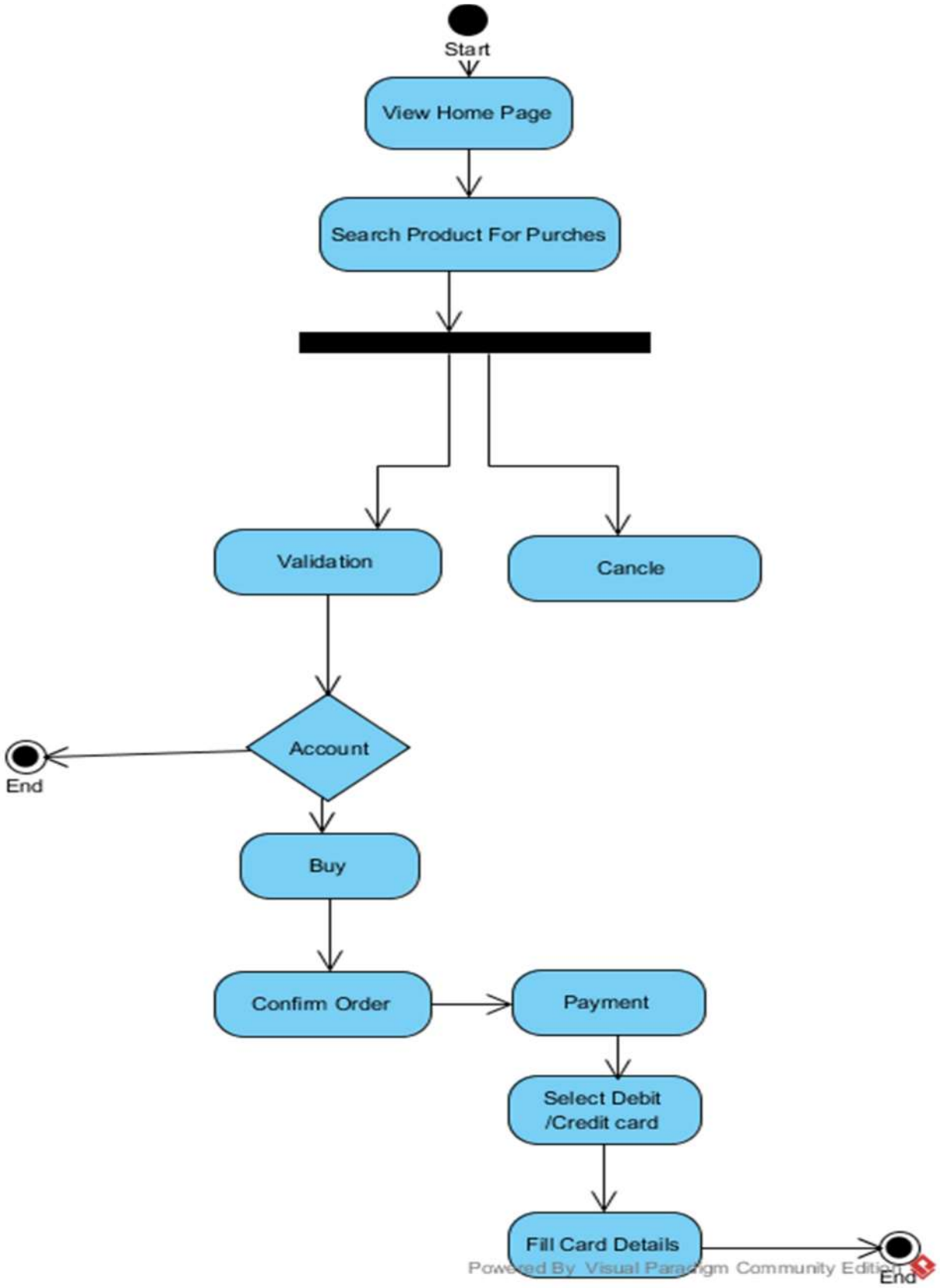
Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows overflow of the control. Activity diagrams are constructed from a limited repertoire of shapes, connected with arrows.

Activity diagrams are constructed from a limited repertoire of shapes, connected with arrows.

## THE MOST IMPORTANT SHAPE TYPES:-

- Rounded rectangle represent activities.
- Diamonds represent decisions.
- Bars represent the start (split) or end (join) of concurrent activities.
- A black circle represents the start (initial state) of the workflow.
- An encircled black circle represents the end (final state)
- Arrows run from the start towards the end and represent the order in which activities happen.

# ACTIVITY DIAGRAM



# SEQUENCE DIAGRAM

A sequence diagram describes interactions among classes in terms of an "Exchange of message over time".

Sequence diagram are used to depict the time sequence of message exchanged between objects.

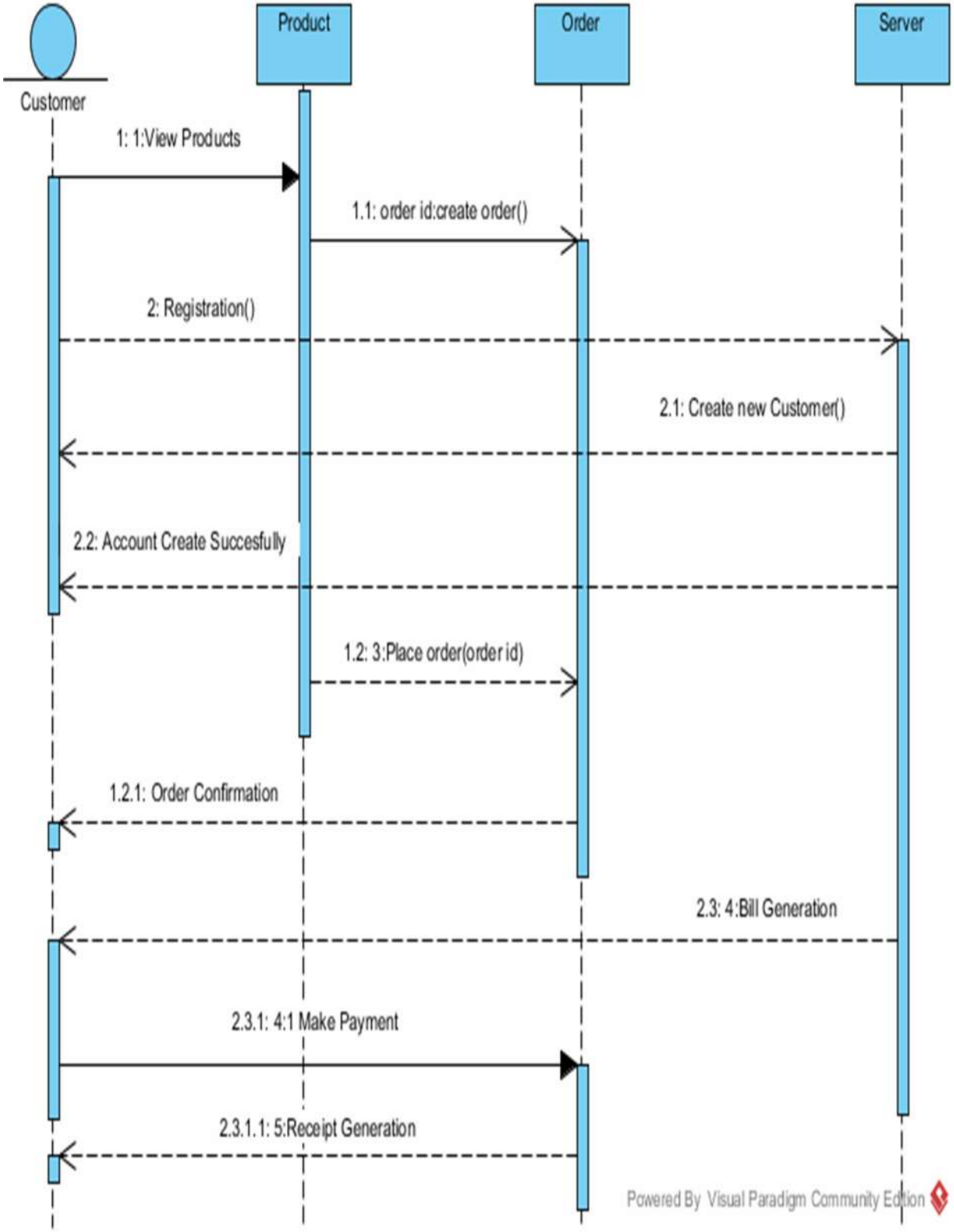
Message can correspond to operation on class or an event trigger.

## Notations of a Sequence Diagram include:

- **Lifeline:-**
  It is vertical dashed line that represents the "lifetime" of an object.
- **Arrows:-**
  They indicate flow of message between objects.

- **Activation:-**
  It is a thin rectangle showing period of time, during which an object is performing an action.

# SEQUENCE DIAGRAM



# STATE CHART DIAGRAM

A state chart diagram describes the dynamic behavior of a system in response to external stimuli.

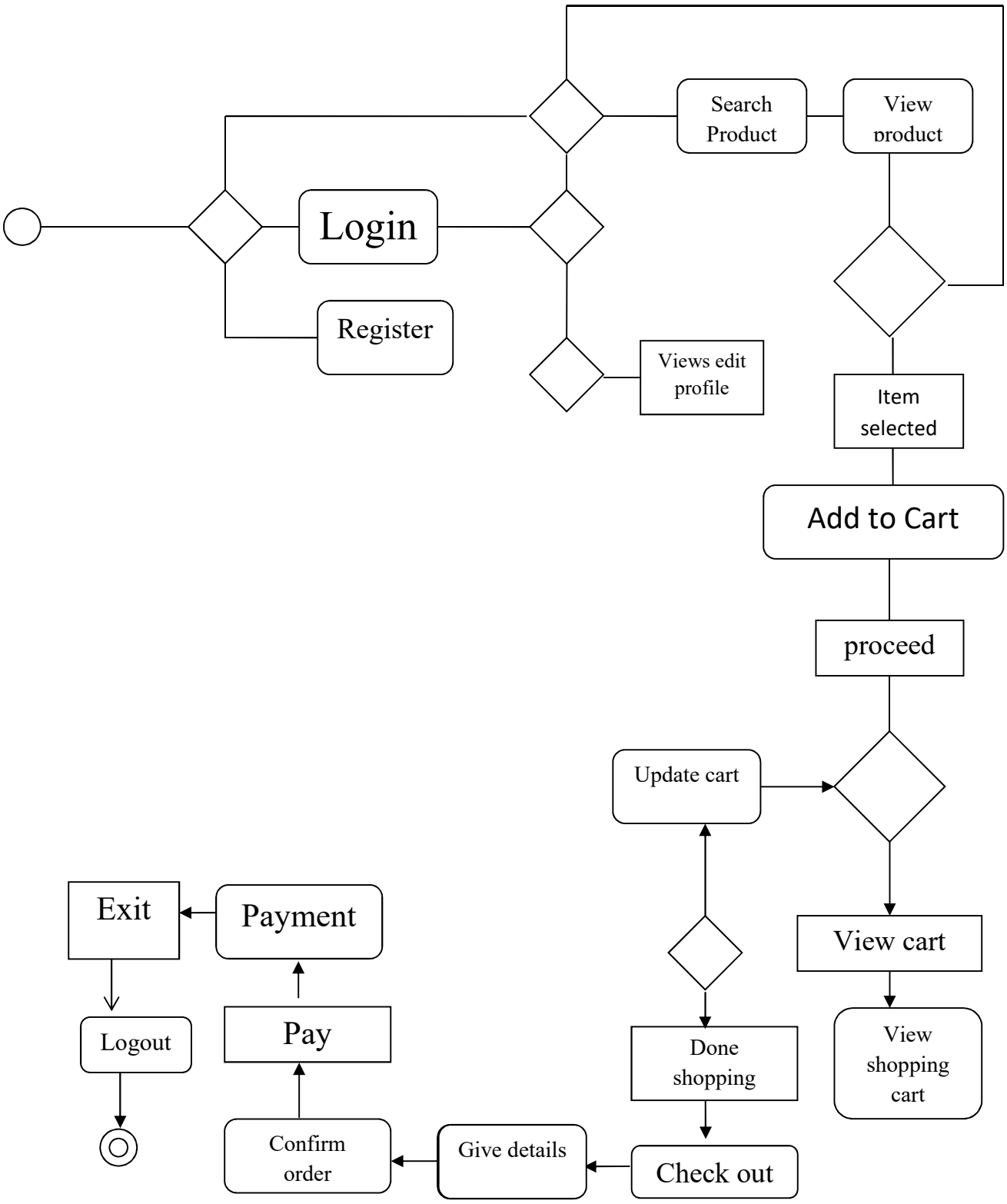
## State Chart Diagram helps:

- To model dynamic behavior of objects based on states..
- To model reactive objects, whose states are triggered by specific events.
- To describe passive objects, which go through several distinct phases during their life time.

## Notations:

- **Name:** Unique name identifying the state.
- **Sub-States:** Set of "disjoints sub-states" or "concurrent sub-states".
- **State-Sub:** Sate relationship: Useful to understand the modeling of complex behaviors.
- **Entry Action:** An action that happens as a result of transition into a state.
- **Exit Action:** An action that happens immediately before a state change.
- **Internal Activity:** Activity that occur within an object while it is in a particular state.

# STATE CHART DIAGRAM



## COMPONENT DIAGRAM

Component Diagram describes the organization of components, including source code, run time(binary) code, and executable.

## Component Diagram:

- Give the physical view of the system in terms of implementation aspect. This is important for reusability and performance purpose.
- Constitute the Components, their interfaces and realizations, and dependencies between components.

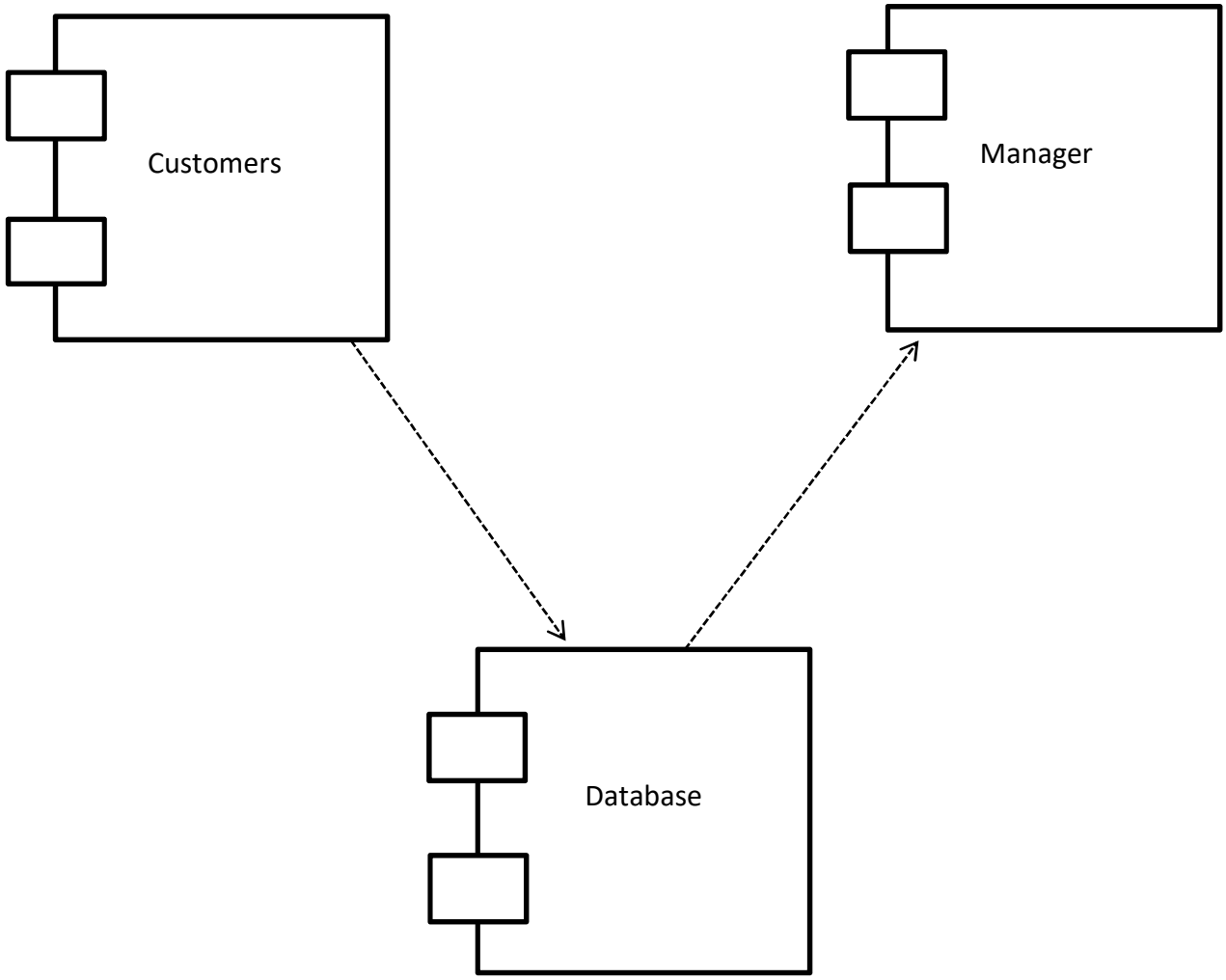
## Component Diagram are used:

- To depict organizations and dependencies among Component type.
- To show allocations of "Classes" and "Objects" to components in the physical designs of the system.
- To indicate the "physical layering" and "partitioning" of the system architecture.

## A component typically encompasses:

- Structure and behavior of a "Collaboration of classes" from the system design.
- Interfaces that describe a group of operations implemented by components.

# COMPONENT DIAGRAM



# DEPLOYMENT DIAGRAM

Deployment Diagram is a structure diagram which shows architecture of the system as deployment (distribution) of software artifacts to deployment targets.

Artifacts represent concrete elements in the physical world that are the result of a development process. Deployment diagrams are usually represented by a node which is either hardware device or some software execution environment. Nodes could be connected through communication paths to create networked systems of arbitrary complexity.

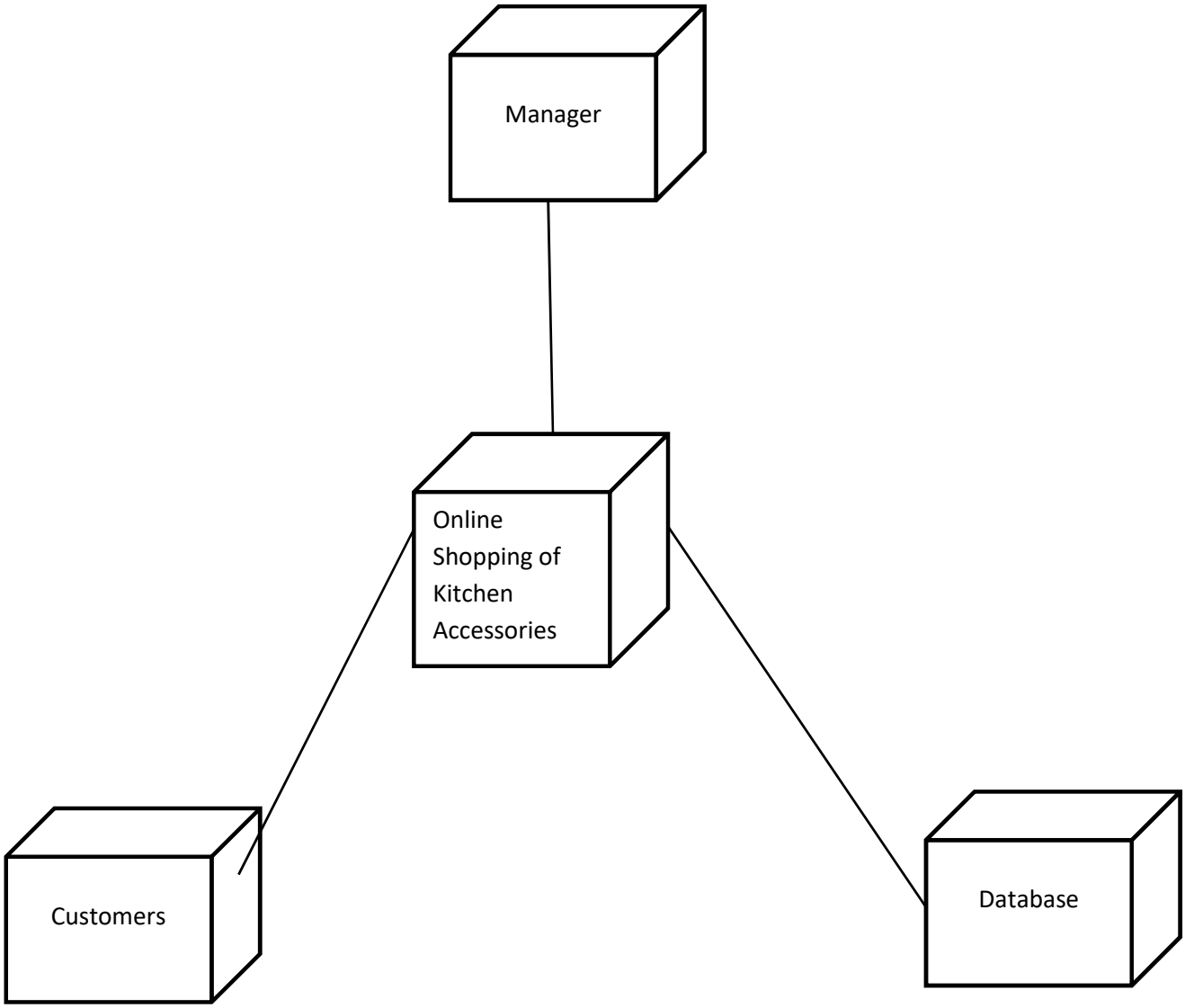
Note, that component was directly deployed to nodes in UML 1.x deployment diagrams. In UML 2.x artifacts are deployed to nodes, and artifacts could manifest (implement) components. Components are deployed to nodes indirectly through artifacts.

Deployment Diagrams could describe architecture at specification level (also called type level) or at instance level (similar to class diagrams and object diagrams).

**Specification level** deployment diagram shows some overview of deployment of artifacts to deployment targets, without referencing specific instances of artifacts or nodes.

**Instance level** deployment diagram shows deployment of instances of artifacts to specific instances of deployment targets. It could be used for example to show the differences in deployments to development, staging or production

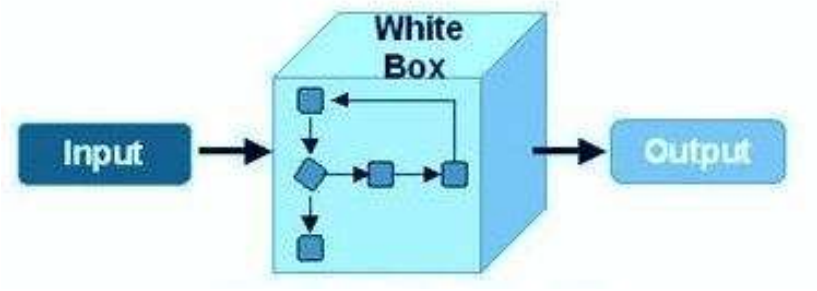
# DEPLOYMENT DIAGRAM



# SYSTEM IMPLLEMENTATION

## WHITE BOX TESTING:

White box testing is a testing technique that examines the program structure and derives test data from the program logic/code. The other names of glass box testing are clear box testing, open box testing, logic driven testing or path driven testing or structural testing.




### Advantages of White Box Testing:

- Forces test developer to reason carefully about implementation.
- Reveals errors in "hidden" code.
- Spots the Dead Code or other issues with respect to best programming practices.

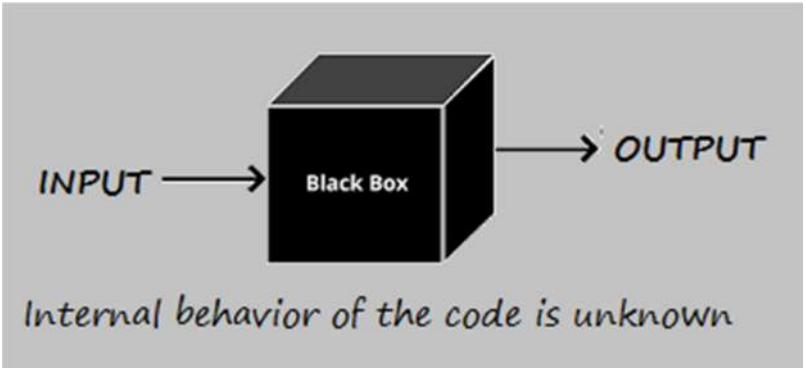
### Disadvantages of White Box Testing:

- Expensive as one has to spend both time and money to perform white box testing.
- Every possibility that few lines of code are missed accidentally.
- In-depth knowledge about the programming language is necessary to perform white box testing.

# BLACK BOX TESTING:

Black-box testing is a method of software testing that examines the functionality of an application based on the specifications. It is also known as Specifications based testing. Independent Testing Team usually performs this type of testing during the software testing life cycle.

This method of test can be applied to each and every level of software testing such as unit, integration, system and acceptance testing.




## Advantages of Black Box Testing

- Unbiased tests because the designer and tester work independently.
- Tester is free from any pressure of knowledge of specific programming languages to test the reliability and functionality of an application / software.
- Facilitates identification of contradictions and vagueness in functional specifications.
- Test is performed from a user's point-of-view and not of the designer's.

## Disadvantages of Black Box Testing

- Tests can be redundant if already run by the software designer
- Test cases are extremely difficult to be designed without clear and concise specifications
- Results might be overestimated at times
- Cannot be used for testing complex segments of code

RESULTS

## RegistrationTable :

### 1.Users:

| Sr.No | Column Name | Data type(size) | Description | Property |
|---|---|---|---|---|
| 1 | Id | int | It Specifies user Id | Primary KeyNot Null |
| 2 | FirstName | varchar(50) | It Specifies First Name of the user | Null |
| 3 | LastName | varchar(50) | It Specifies Last Name of the user | Null |
| 4 | Email | varchar(50) | It Specifies Email of the user | Null |
| 5 | Password | varchar(50) | It specifies Password | Null |
| 6 | City | varchar(50) | It specifies City | Null |
| 7 | Gender | varchar(50) | It Specifies Gender | Null |

## 2.Cookingware:

| Sr.No | Column Name | Data type(size) | Description |
|---|---|---|---|
| 1 | Product_id | Int | It Stores Id |
| 2 | Product_Name | varchar(150) | It Stores Product Name |
| 3 | Product_Images | varchar(150) | It stores the Product Images |
| 4 | Price | Bigint | It Stores the Price |

## 3.Dinningware:

| Sr.No | Column Name | Data type(size) | Description |
|---|---|---|---|
| 1 | Product_id | Int | It Stores Id |
| 2 | Product_Name | varchar(150) | It Stores Product Name |
| 3 | Product_Image | varchar(150) | It stores the Product Images |
| 4 | Price | Bigint | It Stores the Price |

## 4.Orders:

| Sr.No | Column Name | Data type(size) | Description |
|---|---|---|---|
| 1 | Orderid | varchar(50) | It Stores Order Id. |
| 2 | Sno | int | It Stores Serial Number. |
| 3 | productid | bigint | It stores the Product Id. |
| 4 | productname | varchar(150) | It stores the Product Name. |
| 5 | productprice | int | It Stores the Product Price. |
| 6 | quantity | int | It Stores the Product Quantity. |
| 7 | dateoforder | varchar(50) | It Stores the Date Of Order. |
| 8 | address | varchar(50) | It Stores the address of the user. |
| 9 | mobilenumber | varchar(50) | It Store the mobile number. |

## 5.CardDetails:

| Sr.No | Column Name | Data type(size) | Description |
|---|---|---|---|
| 1. | Cardtype | varchar(50) | It Specifies type of card. |
| 2. | cardnumber | varchar(50) | It Specifies card number. |
| 3. | nameoncard | varchar(50) | It Specifies name on card. |
| 4. | Validtill | varchar(50) | It Specifies expiry date of card. |
| 5. | Cvv | int | It Specifies cvv number. |

# FUTURE SCOPE

## Future Enhancement:

- The project made is just to ensure that this product could be valid in today real challenging world. Here all the facilities are made and tested.
- Functionality of overtime handling is kept for future enhancement.
- We look forward to make improvements in this project.

# REFERENCES

## TEST BOOK REFERENCES:


1. Software Engineering. Second Edition Ross Anderson's(Author)
2. Beginning ASP.NET 4.5 IN C# by Matthew MacDonald(Author).


## WEBSITES REFERENCES:


https://www.youtube.com/haritistudyhub..

https://www.c-sharocorner.com

https://drive.google.com/open?id=1ZQh696.../view?usp=drive_open