VNR VIGNANA JYOTI INSTITUTE OF ENGINEERING AND TECHNOLOGY

(Affiliated to J.N.T.U, Hyderabad) Bachupally(v), Hyderabad, Telangana, India

CSBS

II nd Year I Sem

# COURSE BASED PROJECT REPORT

# OBJECT ORIENTED PROGRAMMING LABORATORY

Under the Guidance of

Mrs.Vasavi Ma'am

Asst. Professor, CSE & CSBS

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING &TECHNOLOGY

An Autonomous Institute, ISO9001:2015 & QS I-Gauge Diamond Rated Institute, Accredited by NAAC with 'A++' Grade
NBA Accreditation for B.Tech CE,EEE,ME,ECE,CSE,EIE,IT,AE Programmes
Approved By AICTE, New Delhi, Affiliated to JNTUH, Hyderabad.
Recognized as "College with Potential for Excellence" by UGC

NIRF Ranking: 113 Engineering Category

EAMCET CODE : VJEC
PGECET CODE: VJEC1

Estd. 1995

# CERTIFICATE

*This is to certify that*

21071A3224 – G PRANAV

21071A3245 – M SRAVANTH

21071A3247 – MD ABDUL SAMI

21071A3253 – MANHITH SAI

22075A3205 – KUNDAN KUMAR

*have completed their course project work at CSE Department of VNR VJIET, Hyderabad entitled " " in complete fulfilment of the requirements for the award of B.Tech degree during the academic year 2022-2023. The performance of the students was*

______________________________________

*. and He/she worked well as a part of a Team.*

Signature of the HOD

Signature of the Staff Member

Date of the exam: ______________________

Signature of the examiners

Internal Examiner

External Examiner

# SRS TEMPLATE

**1.INTRODUCTION:** The Billing Management Application for Supermarkets is an application that will be developed for supermarkets to automate their billing process. The application will create a different customer database with their contact number and bill number. The supermarket billing management application will have five categories of items which are added in the bill. The invoice will be generated according to the number of items added and the quantity of each item.
**2.PURPOSE:** The purpose of this software requirement specification (SRS) is to define the requirements of the billing management application for supermarkets. The SRS document will provide the guidelines and requirements to the development team for designing, developing, testing and deploying the application.
3.Scope The scope of the billing management application is to automate the billing process of supermarkets. The application will create a different customer database with their contact number and bill number. The application will have five categories of items which are added in the bill. The invoice will be generated according to the number of items added and the quantity of each item.
4.System Overview The billing management application will be designed to automate the billing process of supermarkets. The application will have a user-friendly interface that will allow the user to enter customer information, add items to the bill, and generate the invoice. The application will have a database to store customer information, bill numbers, and items added to the bill. The application will also have five categories of items which are added to the bill.

**5.FUNCTIONAL REQUIREMENTS :**The functional requirements of the billing management application are as follows:
a. The application should allow the user to enter customer information, such as name, address, and contact number.
b. The application should allow the user to add items to the bill from five different categories.
c. The application should allow the user to enter the quantity of each item added to the bill.
d. The application should generate an invoice according to the number of items added and the quantity of each item.
e. The application should store customer information, bill numbers, and items added to the bill in a database.

**6.NON-FUNCTIONAL REQUIREMENTS :**The non-functional requirements of the billing management application are as follows:
a. The application should have a user-friendly interface.
b. The application should be reliable and accurate.
c. The application should have a fast response time.
d. The application should be secure and protect customer information.

**7.PERFORMANCE REQUIREMENTS:** The performance requirements of the billing management application are as follows:
a. The application should be able to handle multiple users simultaneously.
b. The application should have a fast response time.
c. The application should be able to store large amounts of data.
d. The application should be able to generate an invoice quickly.

**8. SECURITY REQUIREMENTS:** The security requirements of the billing management application are as follows:

a. The application should protect customer information from unauthorized access.

b. The application should encrypt sensitive data, such as customer information and bill numbers.

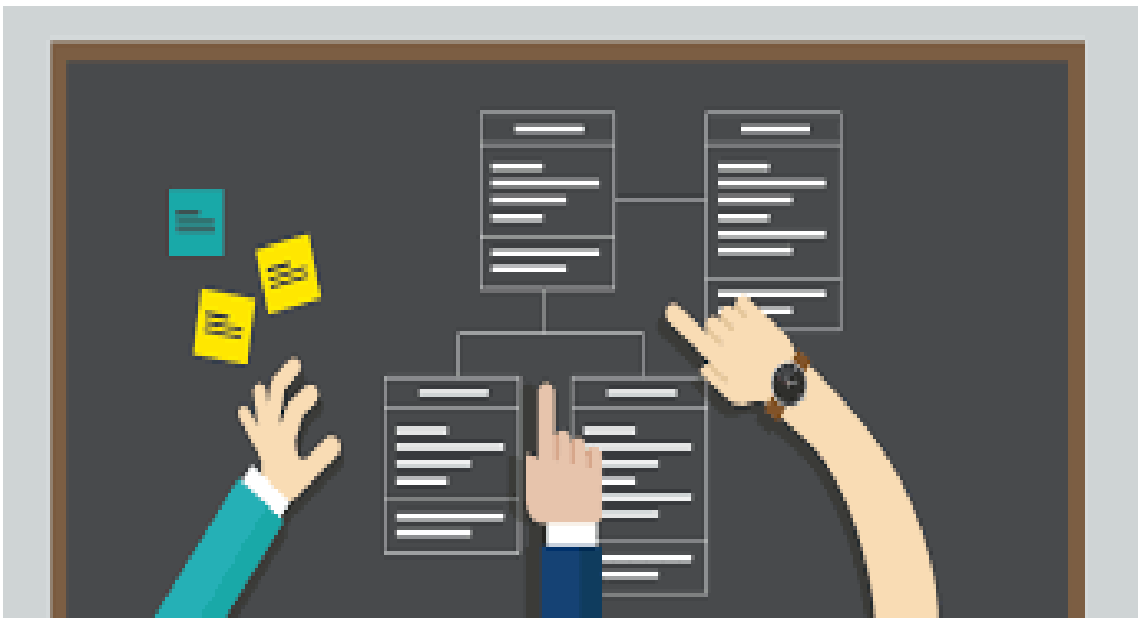
c. The application should have a backup and recovery plan in case of a security breach.

d. The application should have access controls to restrict user access to sensitive data.

**9. CONCLUSION:** The Billing Management Application for Supermarkets is an application that will automate the billing process of supermarkets. The application will create a different customer database with their contact number and bill number. The application will have five categories of items which are added in the bill. The invoice will be generated according to the number of items added and the quantity of each item. This software requirement specification document provides the guidelines and requirements for designing, developing, testing, and deploying the application.

# INTRODUCTION

## Why UML (Unified modelling language) diagrams are so important?



UML (Unified Modeling Language) diagrams are important because they provide a common visual language that can be used by software developers, business analysts, and stakeholders to understand and communicate the design of a system.

UML diagrams are used to represent different aspects of a system, such as its structure, behavior, and interactions. They can be used to model the architecture of a software application, the relationships between objects and classes, the interactions between users and the system, and much more.

The use of UML diagrams enables everyone involved in the development of a system to have a shared understanding of its design, which helps to avoid miscommunication and mistakes. It also allows stakeholders to review and provide feedback on the design before the system is implemented, which can help to identify potential issues and improve the overall quality of the system.

Additionally, UML diagrams can be used to document the design of a system, making it easier for future developers to understand and maintain the code.

## ABSTRACT

There are different types of UML diagrams:
1) Use case diagram
2) Class diagram
3) Sequence diagram
4) Activity diagram 5)Collaboration/Communication diagram
6)Component diagram
7)Deployment diagram
8)Statechart diagram

## USE-CASE DIAGRAM

A Use Case Diagram is a type of UML (Unified Modelling Language) diagram that depicts the relationships between actors and the functions they perform, or use cases.

Use case diagrams provide a high-level view of a system, showing how the system interacts with its users, or actors, to achieve specific goals. In a use case diagram, actors are represented by stick figures and use cases are represented by ovals. The relationship between actors and use cases is shown as lines connecting the actor and the use case.

## CLASS DIAGRAM

A Class Diagram is a type of UML (Unified Modelling Language) diagram that represents the structure of a system by modelling its classes, objects, attributes, and operations.

Class diagrams are used to describe the object-oriented design of a system and provide a graphical representation of the relationships between classes, objects, and their attributes and operations. In a class diagram, classes are represented by rectangles that contain the class name, attributes, and operations. The relationships between classes are represented by various types of lines and arrows, such as inheritance, aggregation, and association.

Class diagrams are used throughout the software development process, from requirements gathering to design, implementation, and maintenance. They help to capture the static structure of a system and to understand the relationships between classes and object.

## SEQUENCE DIAGRAM

A Sequence Diagram is a type of UML (Unified Modelling Language) diagram that represents the interactions between objects in a system over time.

Sequence diagrams are used to model the flow of messages between objects and the order in which they occur. In a sequence diagram, objects are represented by horizontal lines and messages between objects are represented by arrows.
The vertical axis represents time, with messages occurring from top to bottom.

Sequence diagrams are used to describe the dynamic behaviour of a system and provide a graphical representation of the interactions between objects. They are particularly useful for modelling the flow of control in a system and for identifying any potential issues with the interactions between objects.

Sequence diagrams are used throughout the software development process, from requirements gathering to design, implementation, and maintenance. They help to communicate the dynamic behavior of a system and to validate the design before the system is built.

## ACTIVITY DIAGRAM

An activity diagram is a type of UML (Unified Modelling Language) diagram that represents the flow of activities in a system. It is used to model the flow of control in a process, as well as to describe the relationships between activities and the conditions that govern these relationships.

Activity diagrams consist of nodes and edges, where nodes represent activities and edges represent the flow of control between activities. Activities can be actions, decisions, or events. The flow of control is represented by arrows that connect the nodes. The start node is typically represented as a filled circle, while the end node is represented as a bull's eye.

Activity diagrams also support the use of conditional constructs, such as if-then- else statements, which allow for the modelling of decision-making processes. In addition, activity diagrams support the modelling of parallel processing, where multiple activities can be performed simultaneously.

## COLLABORATION/COMMUNICATION DIAGRAM

A Collaboration Diagram, also known as a Communication Diagram, is a type of UML (Unified Modelling Language) diagram that represents the interactions between objects in a system and the relationships between these interactions.

Collaboration diagrams are similar to sequence diagrams, but instead of showing the interactions over time, they show the interactions and relationships between objects at a given point in time. In a collaboration diagram, objects are represented by rectangles and the interactions between objects are represented by arrows. The relationships between objects are indicated by lines connecting the objects.

Collaboration diagrams are used to describe the dynamic behaviour of a system and to model the relationships between objects. They are particularly useful for understanding the relationships between objects and for identifying any potential issues with the interactions between objects.

## COMPONENT DIAGRAM

A Component Diagram is a type of UML (Unified Modelling Language) diagram that represents the structure of a system by modelling its components and the relationships between them.

Component diagrams are used to describe the high-level organization of a system and to show the relationships between its components. In a component diagram, components are represented by rectangles and the relationships between components are represented by lines and arrows.

Components are the building blocks of a system and can be physical or logical components, such as modules, classes, or packages. The relationships between components can represent dependencies, associations, or other relationships.

## DEPLOYMENT DIAGRAM

A Deployment Diagram is a type of UML (Unified Modelling Language) diagram that represents the physical deployment of software components on hardware nodes.

Deployment diagrams are used to describe the physical deployment of a system and to show the relationships between hardware nodes and software components. In a deployment diagram, hardware nodes are represented by nodes, which are connected to software components by lines and arrows.

Deployment diagrams provide a physical view of the system, showing how software components are deployed on hardware nodes. They can also show the relationships between hardware nodes, such as dependencies, associations, or other relationships.

## STATECHART DIAGRAM

A Statechart Diagram, also known as a State Machine Diagram, is a type of UML (Unified Modeling Language) diagram that represents the behavior of an object over time and its possible states.

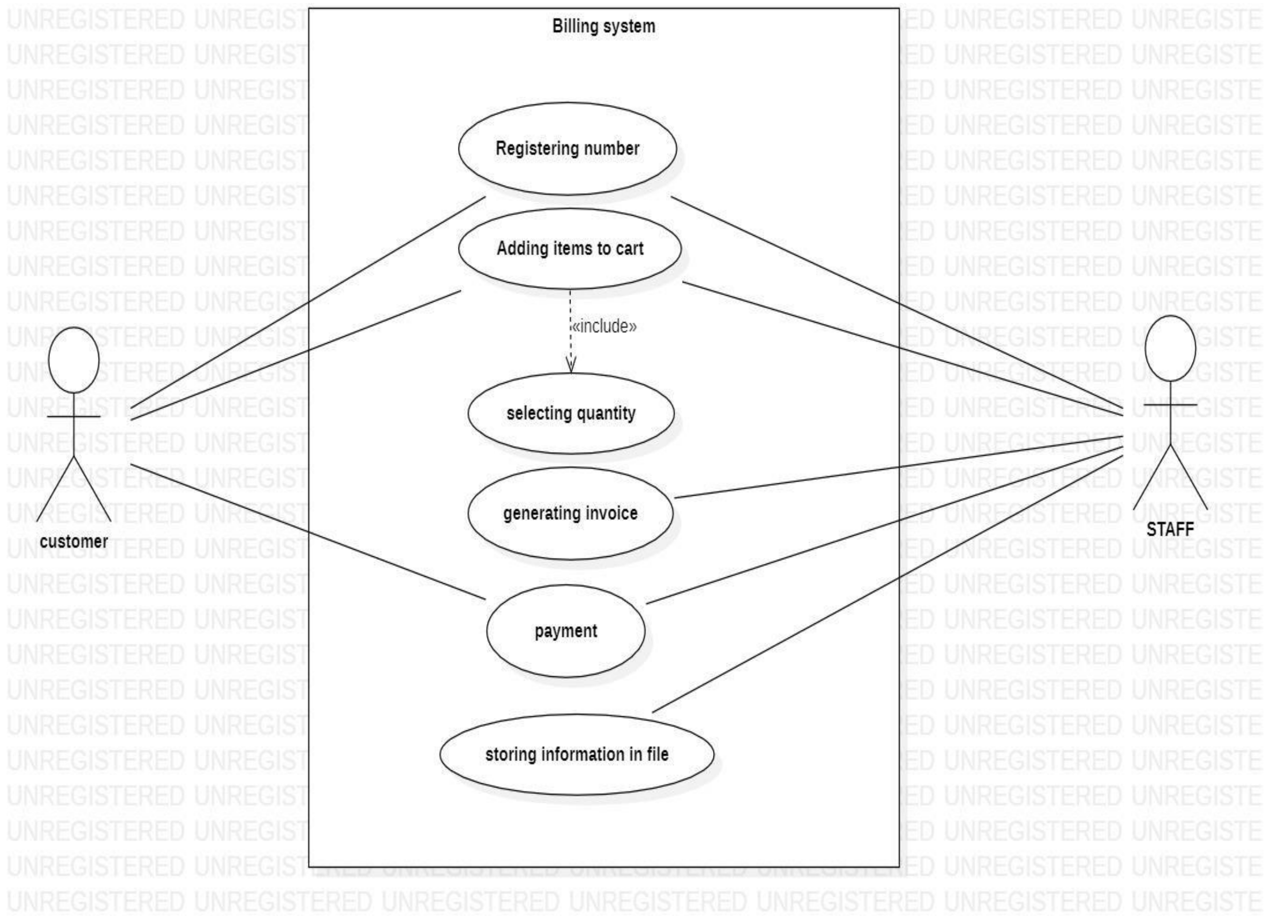
Statechart diagrams are used to describe the dynamic behavior of an object and to model the transitions between its states. In a statechart diagram, states are represented by rectangles and transitions between states are represented by arrows.

Statechart diagrams show the possible states of an object, the events that trigger transitions between states, and the actions that are taken in response to events. They provide a visual representation of the behavior of an object over time and are particularly useful for modeling the dynamic behavior of an object.

Statechart diagrams are used throughout the software development process, from requirements gathering to design, implementation, and maintenance. They help to understand the behavior of an object and to validate the design before the system is built.

# BILLING MANAGEMENT SOFTWARE

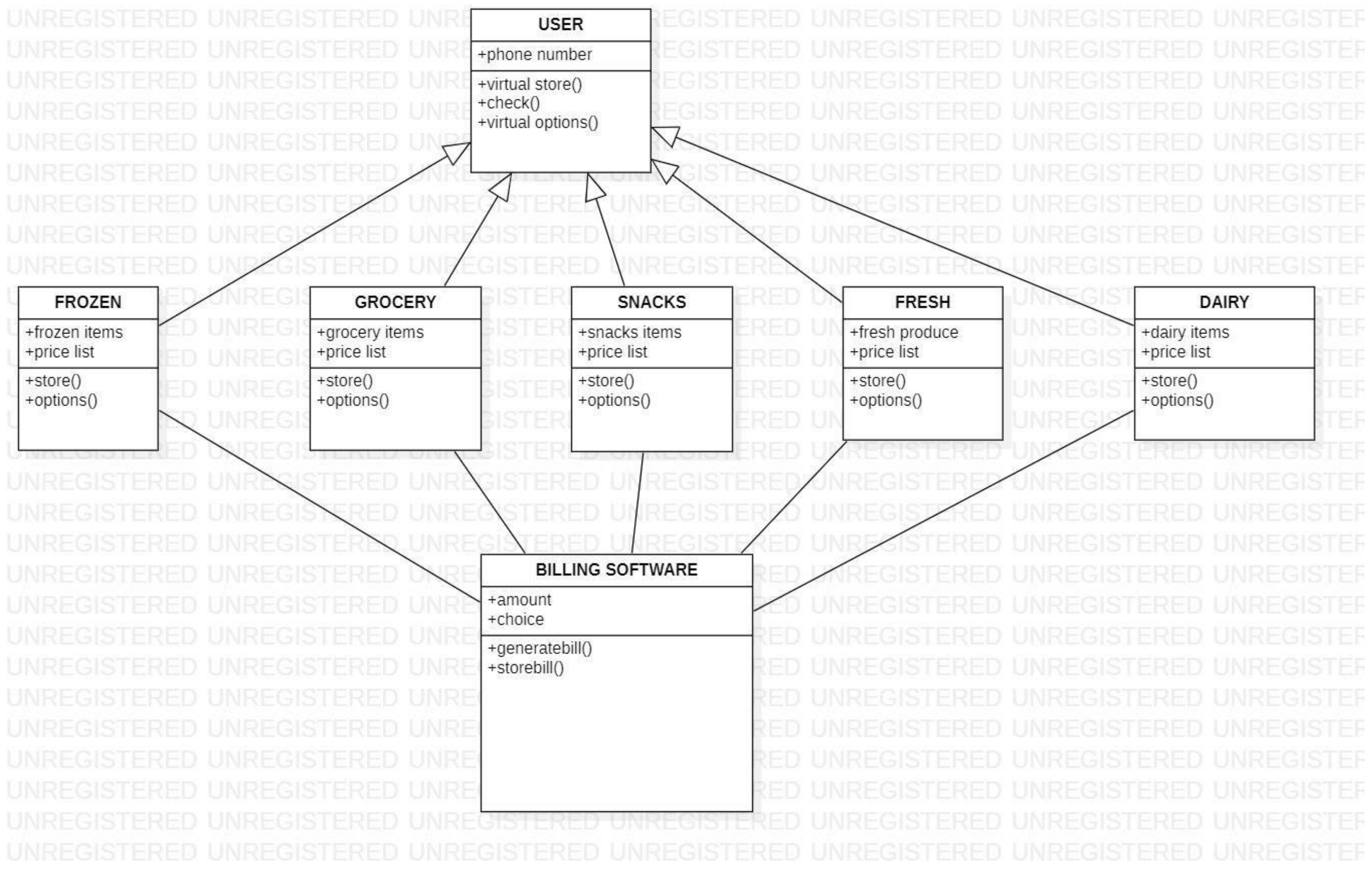
## USE CASE DIAGRAM



Description:

Here we have taken a use case subject named billing system, and 2 actors: customer and staff. The oval shapes in this diagram are the use cases. The use case subject here is billing system. Here, the selecting quantity use case is an included use case of adding items to cart. Because without selecting quantity ,we can't add items to cart. These thick lines in the use case diagram represent association relationship between the actor and the use case.

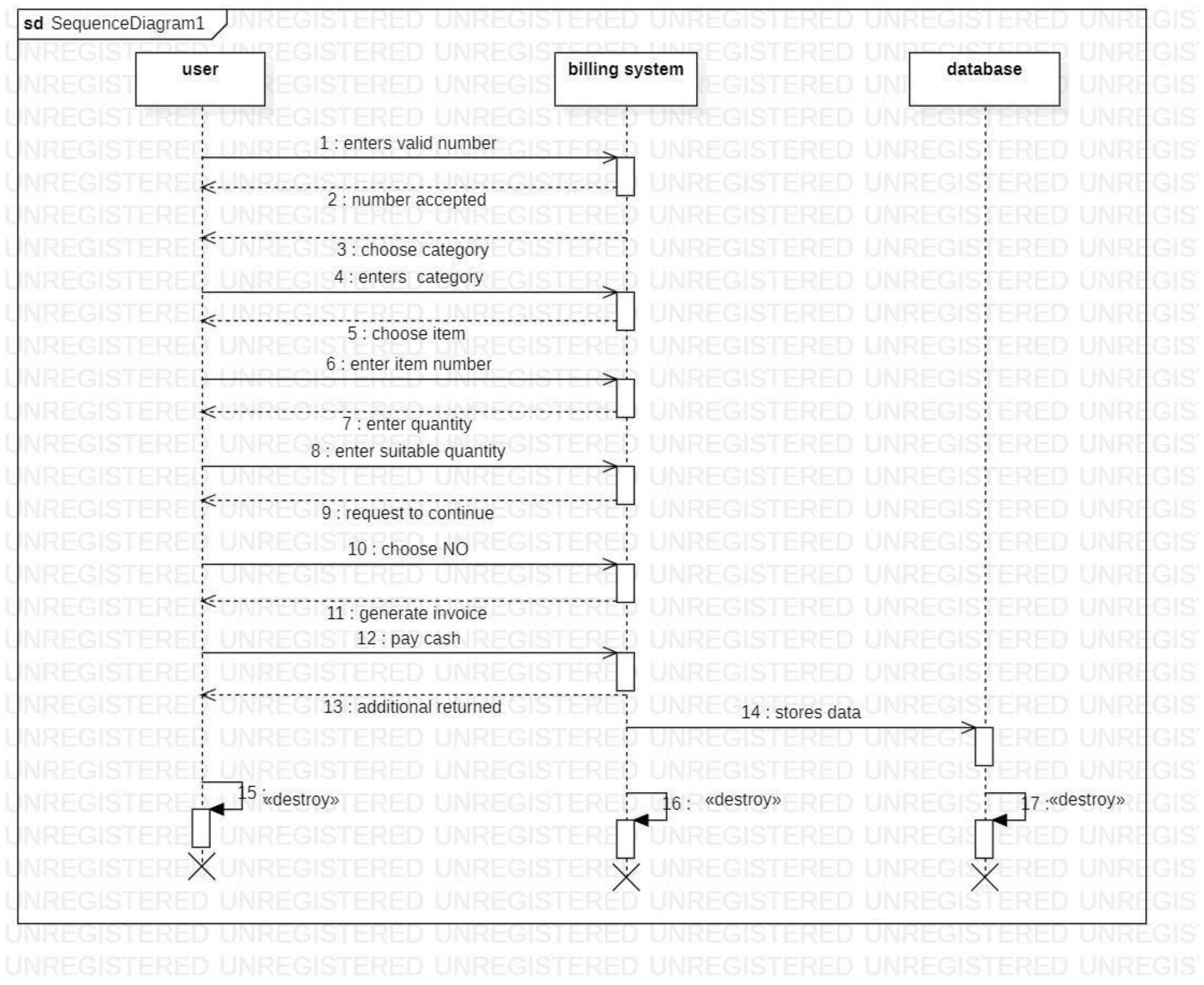
## CLASS DIAGRAM



Description:

Here we have used generalization concept in above the class diagram. The 5 classes here in the diagram, frozen,grocery,snacks,fresh,dairy,are all inherited from the user class.To represent the generalization property, we use a thick line and triangular tip pointing towards the base class(user).After selecting all the items,billing software gathers all this data and generates the bill and stores the bill for future needs. Association(thick line representation) used as a relationship between 2 classes.

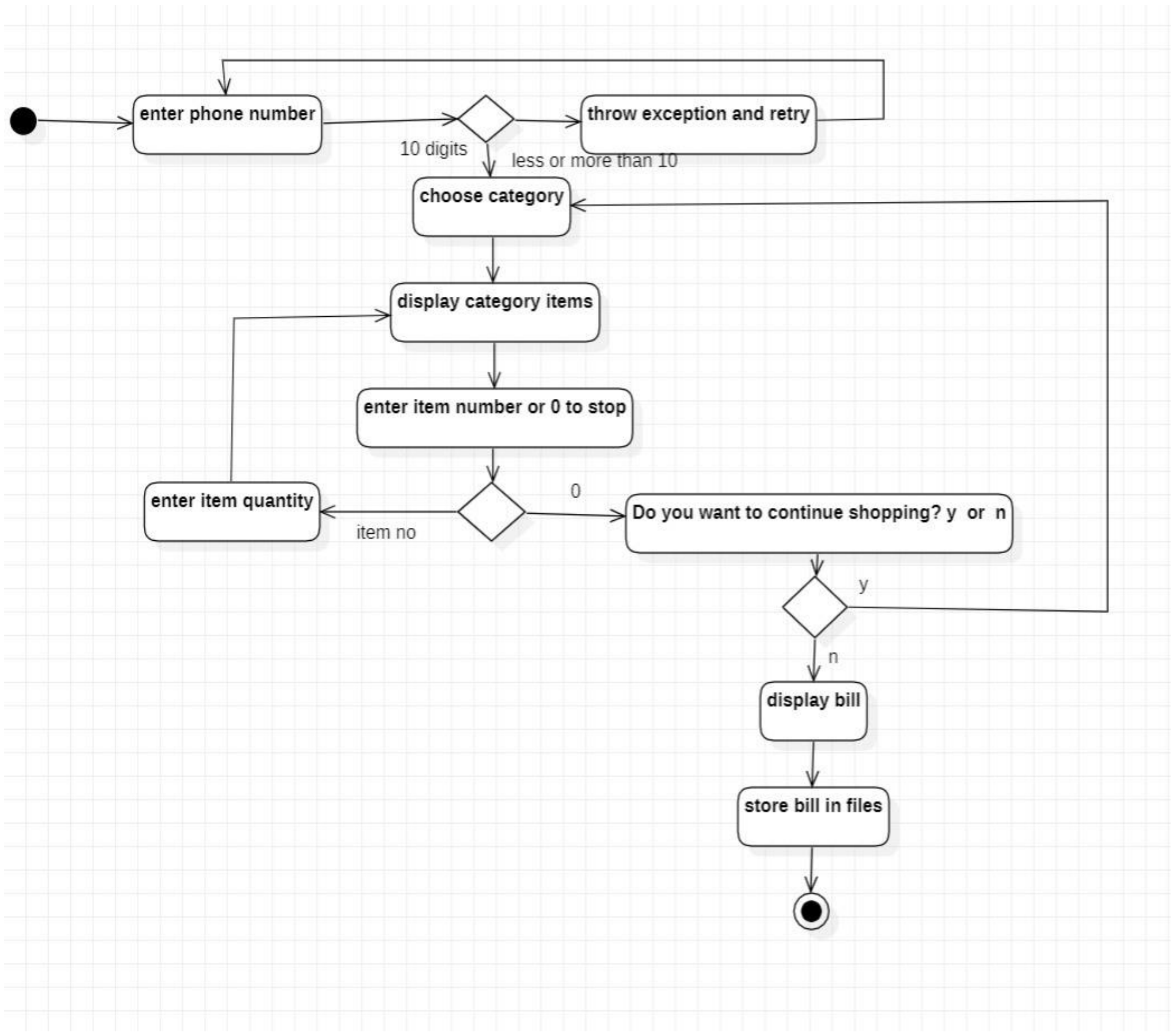
## SEQUENCE DIAGRAM



Description:

The above sequence diagram contains 3 objects(also called as lifelines-user, billing system,database).And the arrowed lines represent messages between 2 objects. The vertical dotted line represents the time
axis,the sequence of messages that happen over a period of time.

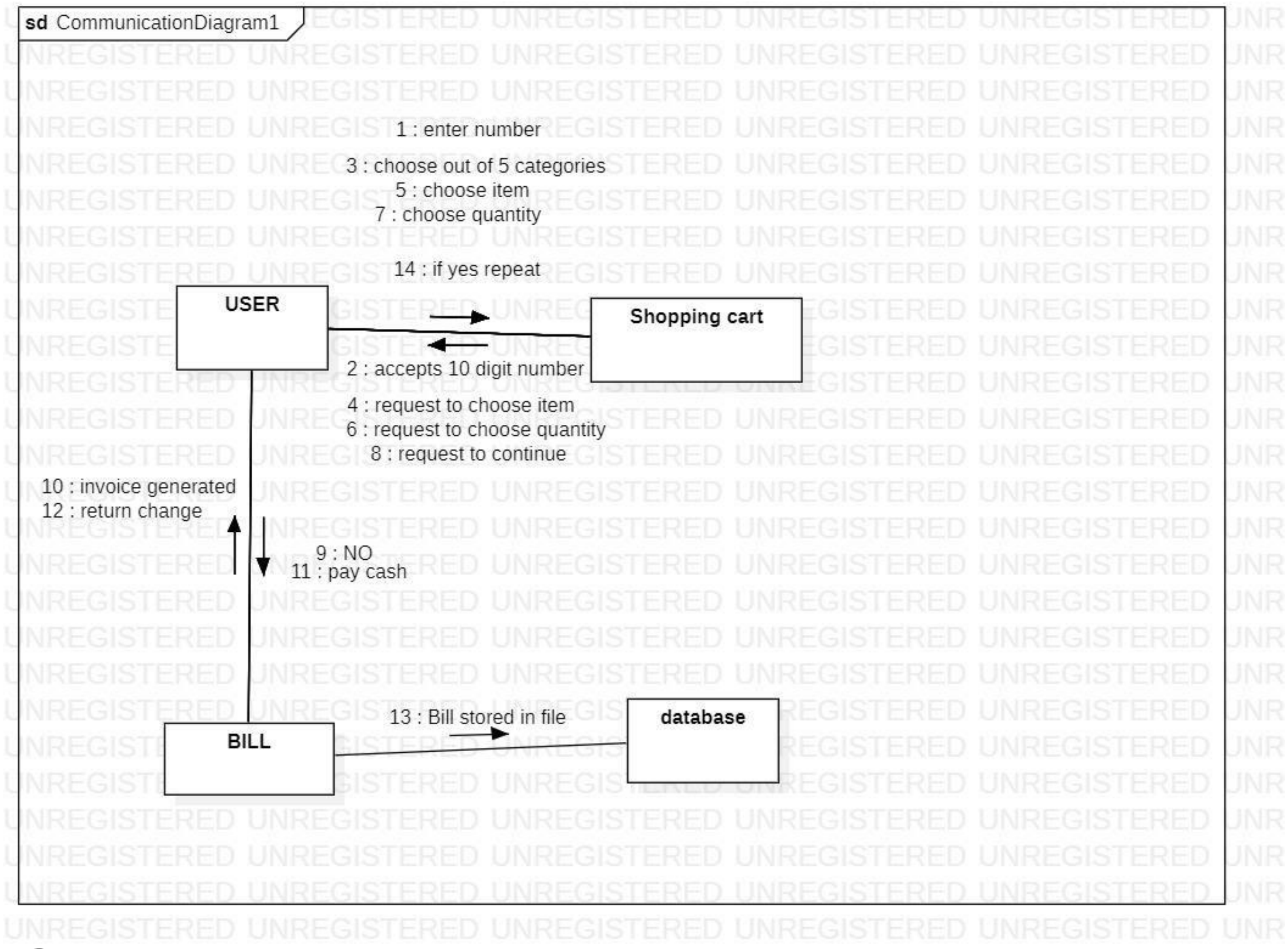
## ACTIVITY DIAGRAM



Description:

Above is an activity diagram representing the flow of actions in the billing management software systems. The initialization symbol(initial) is represented by a filled circle. And the final symbol(final) is represented by a filled circle inside a circle. The actions are represented by the rectangular boxes and decision statements are represented by a rhombus shape,(it has two lines evolving from it-true or false)and arrowed lines represented the action flow .Here in activity diagram , the advantage is that we can handle the exception handling statements which come across in our use case, which will give us more insights .

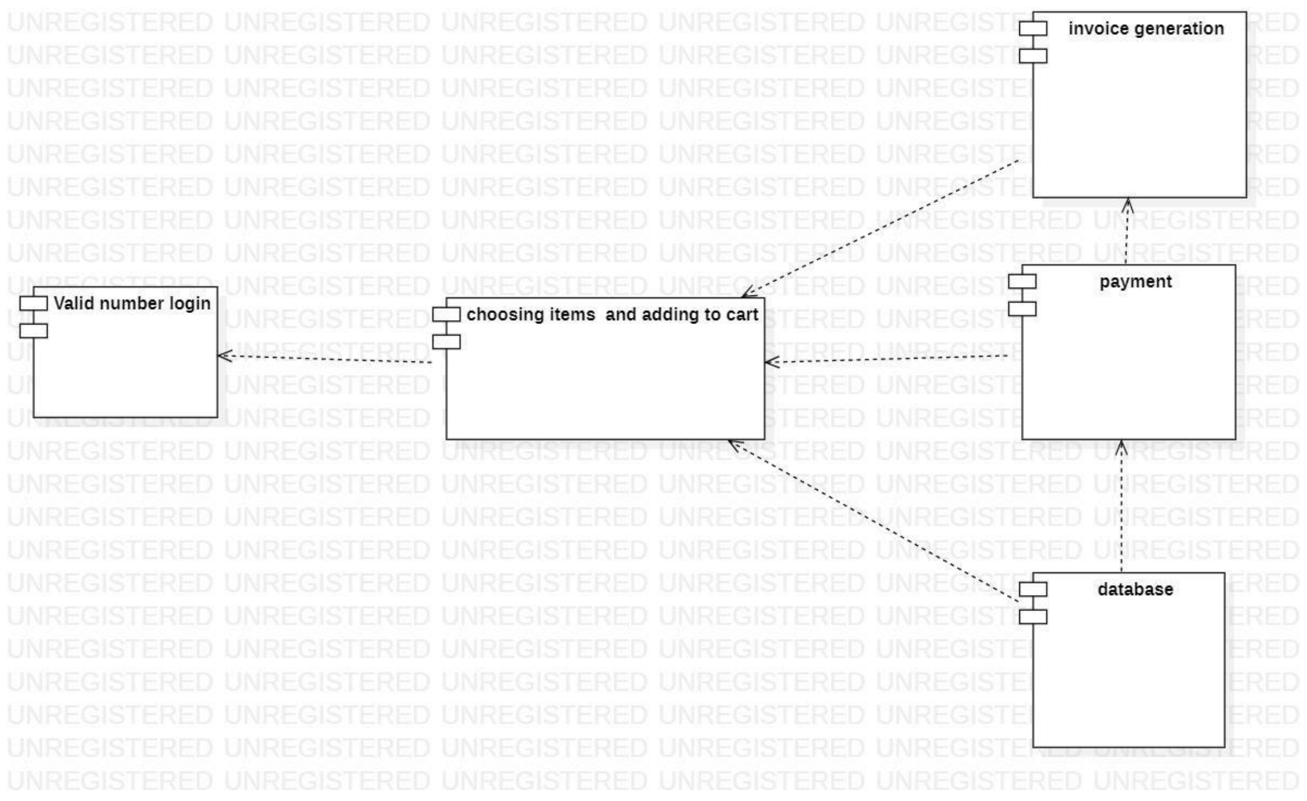
## COMMUNICATION/COLLABORATION DIAGRAM



Description:

In communication diagrams, we have lifelines represent the objects that participate in an interaction. And we have a message pathway called as a connector that identifies objects and passes messages in interaction. As we can see there are connectors between 2 objects and messages are transferred only between those 2 objects, communication is on the whole, but it is important to understand inter-communication between objects ,especially in this use case..(understanding the relation b/w user-bill, user- shopping cart,bill-database)

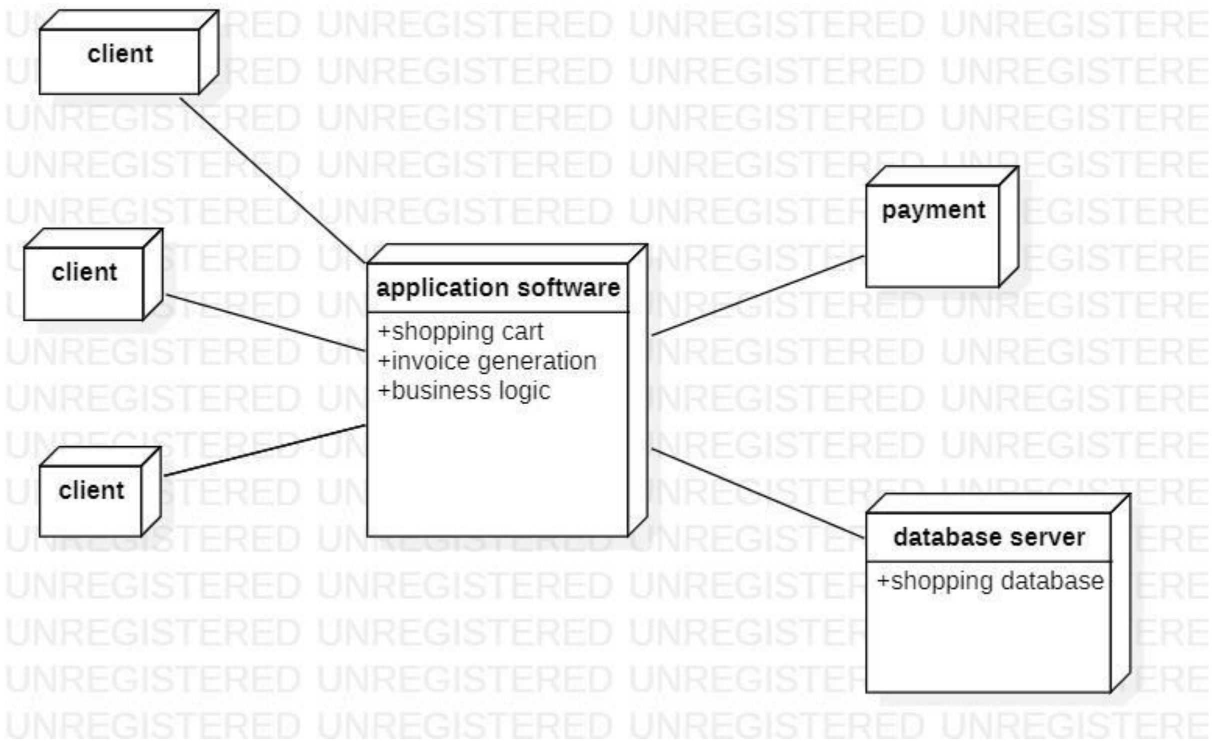
## COMPONENT DIAGRAM



Description:

We have divided our-use case into components (rectangular boxes) and used dependency relationship here.

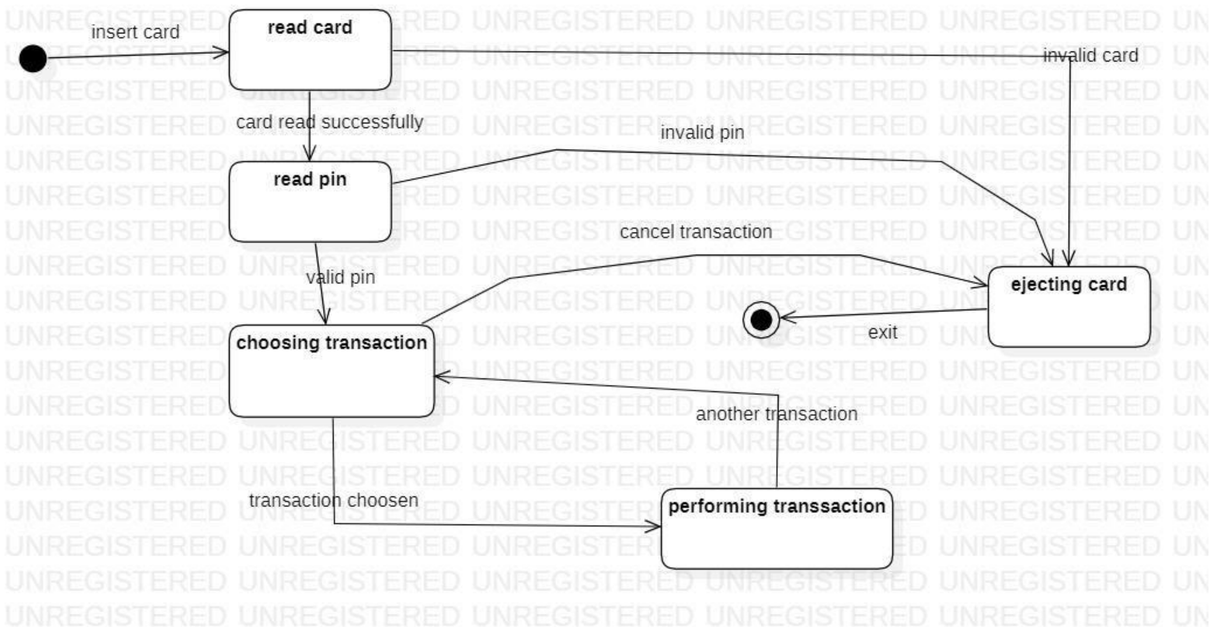
## DEPLOYMENT DIAGRAM



Description:

The deployment diagram of our use case consists of nodes and communication between the nodes using relationships(basically to explain the client-database connections)

## STATECHART DIAGRAM



Description:

Here's the statechart diagram of our use case.We have represented the actions and the transitions between 2 objects.

References:

www.google.com
www.geeksforgeeks.com
www.javatpoint.com

★ ★