ONLINE SHOPPING SYSTEM

PROJECT REPORT

18CSC202J/ 18AIC203J - OBJECT ORIENTED DESIGN AND PROGRAMMING LABORATORY

(2018 Regulation)

II Year/ III Semester

Academic Year: 2022 -2023

By

SAMSON ZACHARIA (RA2111003010220)

DR GUNA SHRESHT (RA2111003010231)

SHRESTHA ROY (RA2111003010234)

SIVANI ANBUSELVAN (RA2111003010235)

Under the guidance of

Dr. G. PADMAPRIYA

Assistant Professor

Department of Computational Intelligence

FACULTY OF ENGINEERING AND TECHNOLOGY

SCHOOL OF COMPUTING

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Kattankulathur, Kancheepuram

NOVEMBER 2022

BONAFIDE

This is to certify that **18CSC202J - OBJECT ORIENTED DESIGN AND PROGRAMMING LABORATORY project report** titled **"ONLINE SHOPPING SYSTEM"** is the bonafide work of who undertook the task of completing the project within the allotted time.

Signature of the Guide	**Signature of the II Year Academic Advisor**
Dr. S. Amudha	----------
Assistant Professor	**Professor and Head**
Department of CINTEL,	Department of CINTEL
SRM Institute of Science and Technology	SRM Institute of Science and Technology

About the course:-

18CSC202J/ 8AIC203J - Object Oriented Design and Programming are 4 credit courses with **L T P C as 3-0-2-4** (Tutorial modified as Practical from 2018 Curriculum onwards)

Objectives:

The student should be made to:

- Learn the basics of OOP concepts in C++
- Learn the basics of OOP analysis and design skills.
- Be exposed to the UML design diagrams.
- Be familiar with the various testing techniques

Course Learning Rationale (CLR): The purpose of learning this course is to:

1. Utilize class and build domain model for real-time programs
2. Utilize method overloading and operator overloading for real-time application development programs
3. Utilize inline, friend and virtual functions and create application development programs
4. Utilize exceptional handling and collections for real-time object-oriented programming applications
5. Construct UML component diagram and deployment diagram for design of applications
6. Create programs using object-oriented approach and design methodologies for real-time application development

Course Learning Outcomes (CLO): At the end of this course, learners will be able to:

1. Identify the class and build domain model
2. Construct programs using method overloading and operator overloading
3. Create programs using inline, friend and virtual functions, construct programs using standard templates
4. Construct programs using exceptional handling and collections
5. Create UML component diagram and deployment diagram
6. Create programs using object oriented approach and design methodologies

Table 1: Rubrics for Laboratory Exercises

(Internal Mark Splitup:- As per Curriculum)

CLAP-1	5=(2(E-lab Completion) + 2(Simple Exercises)(from CodeZinger, and any other coding platform) + 1(HackerRank/Code chef/LeetCode Weekend Challenge)	Elab test
CLAP-2	7.5=(2.0(E-lab Completion)+ 2.0 (Simple Exercises)(from CodeZinger, and any other coding platform) + 3.5 (HackerRank/Code chef/LeetCode Weekend Challenge)	Elab test
CLAP-3	7.5=(2.0(E-lab Completion(80 Pgms)+ 2.0 (Simple Exercises)(from CodeZinger, and any other coding platform) + 3.5 (HackerRank/Code chef/LeetCode Weekend Challenge)	**2 Mark -** E-lab Completion **80 Program** Completion from 10 Session (Each session min 8 program) **2 Mark -** Code to UML conversion GCR Exercises **3.5 Mark - Hacker Rank** Coding challenge completion
CLAP-4	5= 3 (Model Practical) + 2(Oral Viva)	• **3 Mark** – Model Test • **2 Mark** – Oral Viva
Total	25	

COURSE ASSESSMENT PLAN FOR OODP LAB

S.No	List of Experiments	Course Learning Outcomes (CLO)	Blooms Level	PI	No of Programs in each session
1.	Implementation of I/O Operations in C++	CLO-1	Understand	2.8.1	10
2.	Implementation of Classes and Objects in C++	CLO-1	Apply	2.6.1	10
3,	To develop a problem statement. 1. From the problem statement, Identify Use Cases and develop the Use Case model. 2. From the problem statement, Identify the conceptual classes and develop a domain model with a UML Class diagram.	CLO-1	Analysis	4.6.1	Mini Project Given
4.	Implementation of Constructor Overloading and Method Overloading in C++	CLO-2	Apply	2.6.1	10
5.	Implementation of Operator Overloading in C++	CLO-2	Apply	2.6.1	10
6.	Using the identified scenarios, find the interaction between objects and represent them using UML Sequence diagrams and Collaboration diagrams	CLO-2	Analysis	4.6.1	Mini Project Given
7.	Implementation of Inheritance concepts in C++	CLO-3	Apply	2.6.1	10
8.	Implementation of Virtual function & interface concepts in C++	CLO-3	Apply	2.6.1	10
9.	Using the identified scenarios in your project, draw relevant state charts and activity diagrams.	CLO-3	Analysis	4.6.1	Mini Project Given
10.	Implementation of Templates in C++	CLO-3	Apply	2.6.1	10
11.	Implementation of Exception of Handling in C++	CLO-4	Apply	2.6.1	10
12.	Identify the User Interface, Domain objects, and Technical Services. Draw the partial layered, logical architecture diagram with UML package diagram notation such as Component Diagram, Deployment Diagram.	CLO-5	Analysis	4.6.1	Mini Project Given
13.	Implementation of STL Containers in C++	CLO-6	Apply	2.6.1	10
14.	Implementation of STL associate containers and algorithms in C++	CLO-6	Apply	2.6.1	10
15.	Implementation of Streams and File Handling in C++	CLO-6	Apply	2.6.1	10

LIST OF EXPERIMNENTS FOR UML DESIGN AND MODELLING:

To develop a mini-project by following the exercises listed below.

1. To develop a problem statement.
2. Identify Use Cases and develop the Use Case model.
3. Identify the conceptual classes and develop a domain model with UML Class diagram.
4. Using the identified scenarios, find the interaction between objects and represent them using UML Sequence diagrams.
5. Draw relevant state charts and activity diagrams.
6. Identify the User Interface, Domain objects, and Technical services. Draw the partial layered, logical architecture diagram with UML package diagram notation.

Suggested Software Tools for UML:

StarUML, Rational Suite, Argo UML (or) equivalent, Eclipse IDE and Junit

ABSTRACT

An online shopping system is a process in which people (specifical customers) are being provided with the option of purchasing goods and services directly from the seller, all in a real-time environment. Online shopping is an application of the internet as electronic commerce. This system involves its own database to be maintained. As the information or details about the products are stored in the database (like RDBMS, online databases on a paid basis like firebase, etc.) for the server-side functionalities. The Server process is for dealing with the customer's detail and the items that are shipped to different locations based on the addresses provided by the customers. The application design contains two modules one is for the customers who wish to buy the articles. And another is for the store owners who maintain and updates the information regarding the articles and about the customers. The end-user to use this product are the common people for whom the application is to be hosted on the web and the admin maintains the database. The application that is deployed on the customer's database like RDBMS, the information regarding the items is highlighted and forwarded from the database for the customer (front view) based on the choice through the menu list and based on all these searches and transactions the database of all the products is updated at the end of each transaction. The entries for products, into the application, can be made through various screens designed for various levels of users. As soon as, the authorized personnel feeds the relevant data into the system, several reports are generated based on the security policy used.

Online shopping System UML Diagrams are used to represent the shopping system as well as its primary users, roles, activities, artifacts, or classes.

MODULE DESCRIPTION

MODULES:

The online shopping system after careful analysis has been identified to be presented with the following modules and roles.

The modules involved are:

- Administrator
- Moderators
- Users

ADMINISTRATOR:

The administrator is the super user of this application. Only admin have access into this admin page. Admin may be the owner of the shop.

The administrator has all the information about all the users and about all products.

This module is divided into different sub-modules.

1. Manage Moderators
2. Manage Products
3. Manage Users
4. Manage Orders

MODERATORS:

A moderator is considered as a staff who can manage orders for the time being. As a future update moderator may give facility to add and manage his own products .

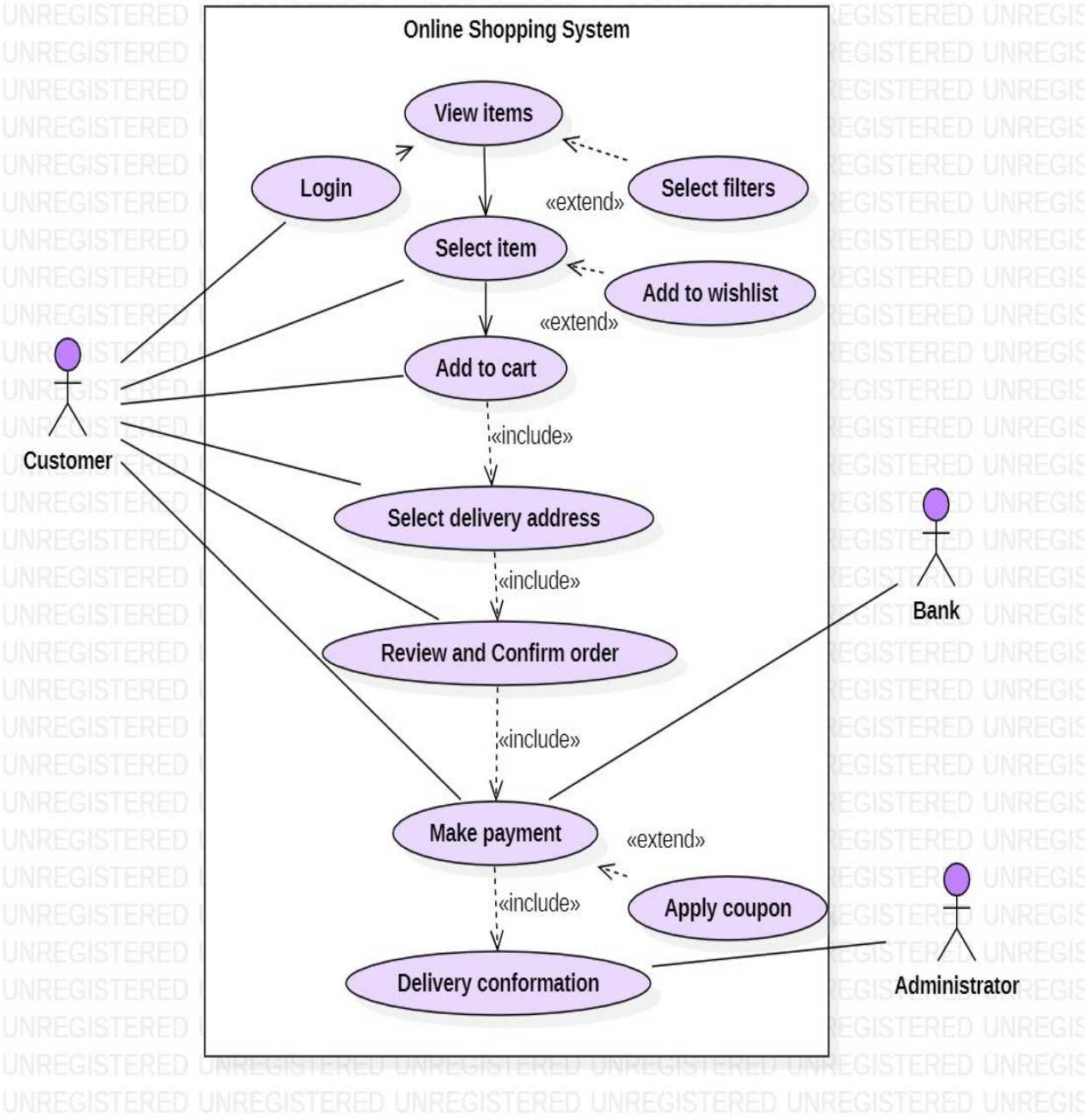
He can add products and users. He can also check the orders and edit his profile.

- Manage products
- Manage users
- Manage orders

USERS:

A new user will have to register in the system by providing essential details in order to view the products in the system. The admin must accept a new user by unblocking him.

USE CASE DIAGRAM



The use case diagram for online shopping system project is a way to show how shoppers and the online shopping system talk to each other. It shows how the system is put together and how it works. The UML use case diagram for online shopping system shows the sample behavior diagram of the software. It includes the project functions using use cases, actors, and their connections. Use-case diagrams illustrate and define the context and requirements of either the

entire online shopping system or the important parts of the system. We can model a complex system with a single use-case diagram, or create many use-case diagrams to model the components of the system. We would typically develop use-case diagrams in the early phases of a project and refer to them throughout the development process.

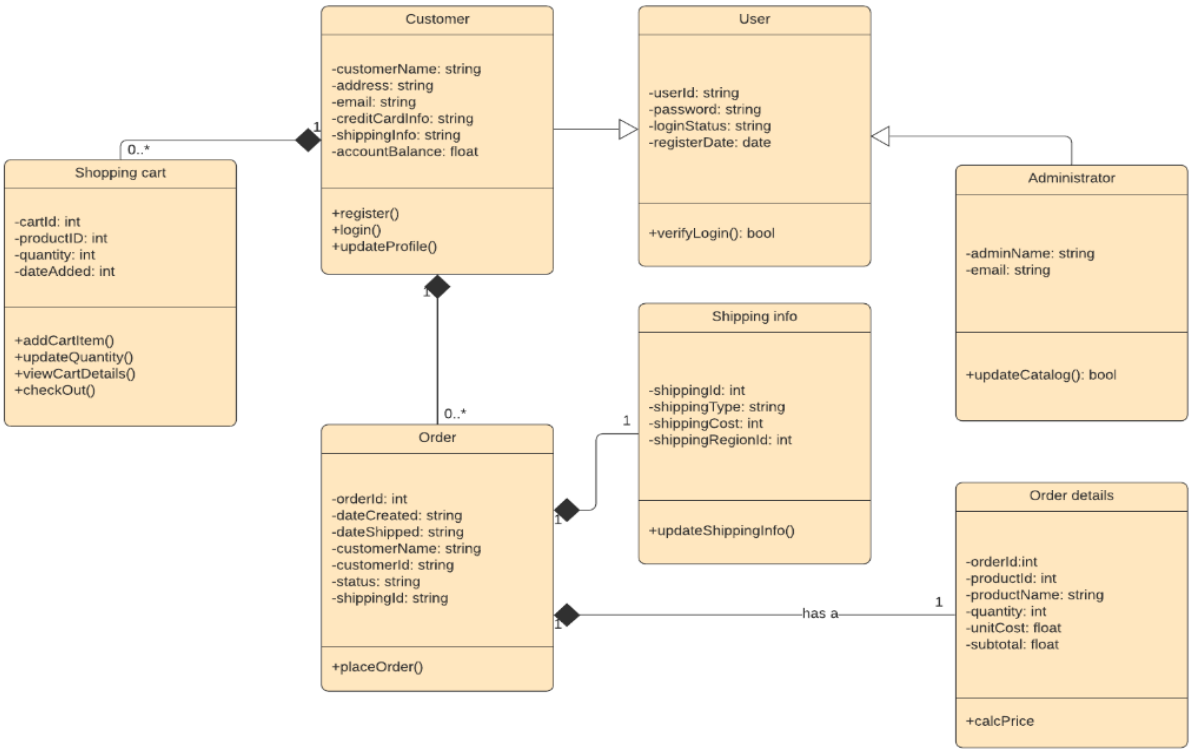
The use case diagram using include and extend is used to elaborate on the preceding diagrams. “include” means that the use cases are needed to finish the task, while “extend” means that they are not.

Actor uses some web site to make purchases online. Top level use cases are View Items. View Items use case could be used by customer as top level use case if customer only wants to find and see some products. This use case could also be used as a part of Make Purchase use case. View items use case is extended by several optional use cases - customer may search for items, view items recommended for him/her, add items to shopping cart or wish list. All these use cases are extending use cases because they provide some optional functions allowing customer to find item.

This use case explains how the admin categorizes products and displays them on the customer’s interface. This action saves seller product info. Update product status. Numbering customer orders and deliveries. Orders and delivery management includes client requests and orders through deliveries. Date of reservation or order, number and price of items, total amount paid, and delivery date.

Therefore, the use case diagram for online shopping system pdf represents the methodology used in system development. It helps developers know the possible inputs that the project should process and perform.

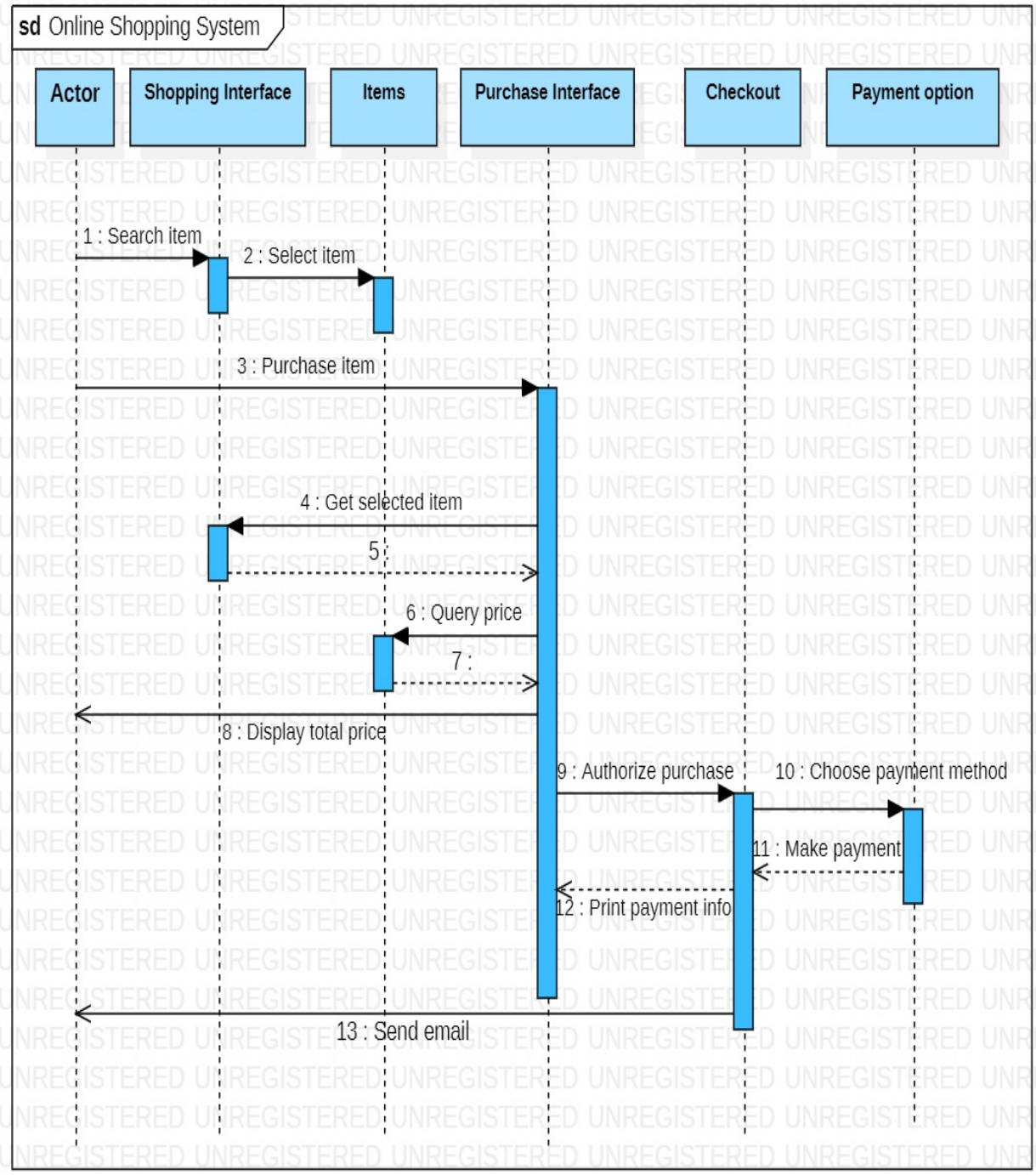
CLASS DIAGRAM



The Online Shopping System Class Diagram is a designed diagram that shows the system's relationships and classes. It contains the class attributes, methods as well as the relationships between classes. These mentioned contents make sure that your Online Shopping system development must in line with what should be its functions. So the classes that must be made in an Online Shopping are the users, sellers, products, order information, deliveries, and transaction.

Class diagrams are used in software engineering to describe the structure of a system. A class diagram uses Unified Modeling Language (UML) to show the classes, attributes, methods (or operations), and their relationships to each other in the system. Class diagrams prove valuable in object-oriented modeling. Class diagrams can be used to model the data structure or to design a system in detail.

SEQUENCE DIAGRAM



Making a purchase online involves a lot of back and forth between the user, the store, the payment gateway, and even potentially a bank, if a transfer is

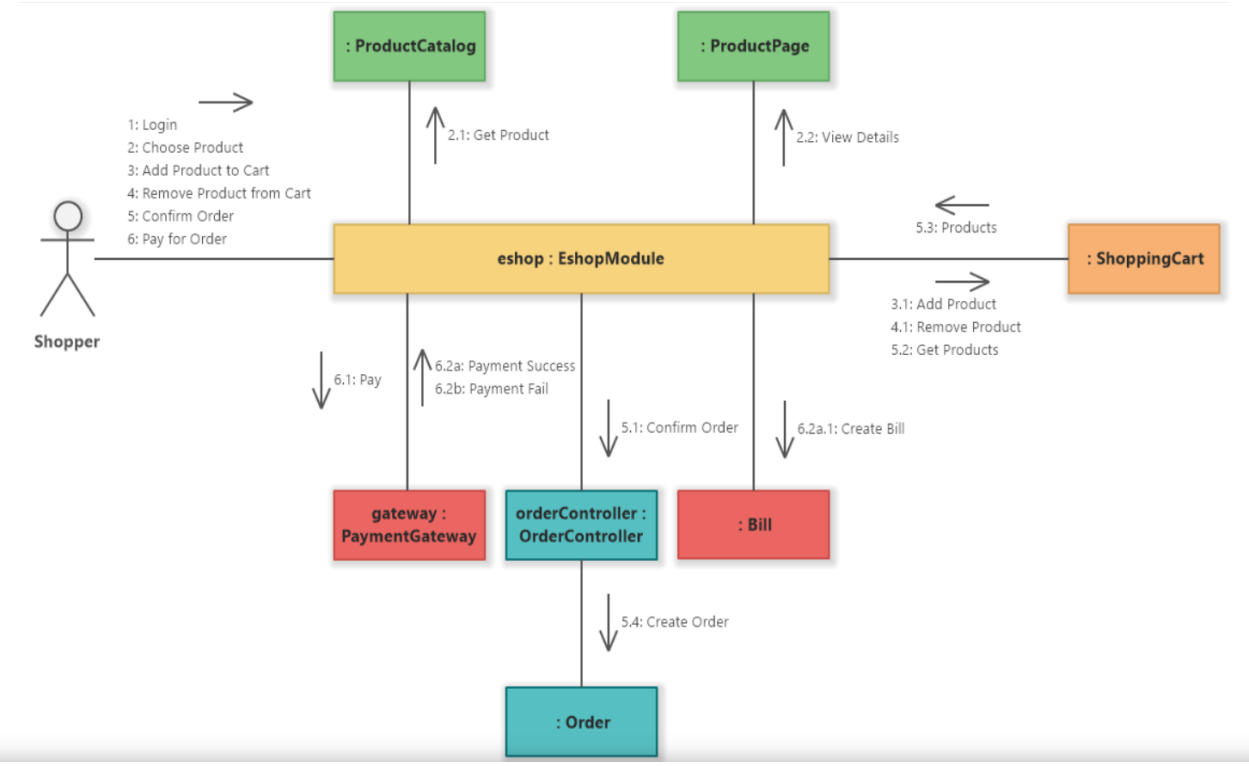
involved. That makes this kind of system perfect for describing with a sequence diagram.

A sequence diagram captures messaging between objects in a system and indicates which object is active at any particular point in the process of carrying out operations. Sequence diagrams help a software engineer or developer visualize the order in which events occur and how they trigger activity within the system.

Sequence diagrams use a horizontal axis for the objects being modeled, with the vertical axis showing how time progresses from top to bottom. Objects are activated as messages travel back and forth on the horizontal axis. These messages are shown as arrows that can go both left and right. Each object has a lifeline that descends vertically and this lifeline will show an activation bar if the object is active.

Because we're modeling a sequence in time, we'll dive straight in and start at the point where a customer has decided on their purchase. The first step is to add an item to the shopping cart. We'll need two objects – the user and the online store. Now we can start to involve the other objects that we need in our sequence diagram, the payment gateway (for instance, PayPal or Stripe), and the bank. The online store then needs to communicate with the payment gateway. This lets the user pay with their card. The online store has to give the user the bank details they need to make the transfer. This message will therefore go in the opposite direction from the previous messages and go back to the user. This is known as a reply message and is shown with a dotted line with an arrow. The user can then go ahead and complete the transfer. This will send a message to their bank. The final stage in the purchasing process is that the online store sends a confirmation message to the user and this ends the part of the sequence diagram that we want to model.

COMMUNICATION DIAGRAM



A communication diagram for online shopping system is an extension of object diagram that shows the objects along with the messages that travel from one to another. In addition to the associations among objects, communication diagram shows the messages the objects send each other.

Communication Lifelines:

Lifelines represent modules and system parts needed in the online shopping process within modeled example e-shop solution:

- Shopper
- Eshop Module
- Product Catalog
- Product Page
- Shopping Cart

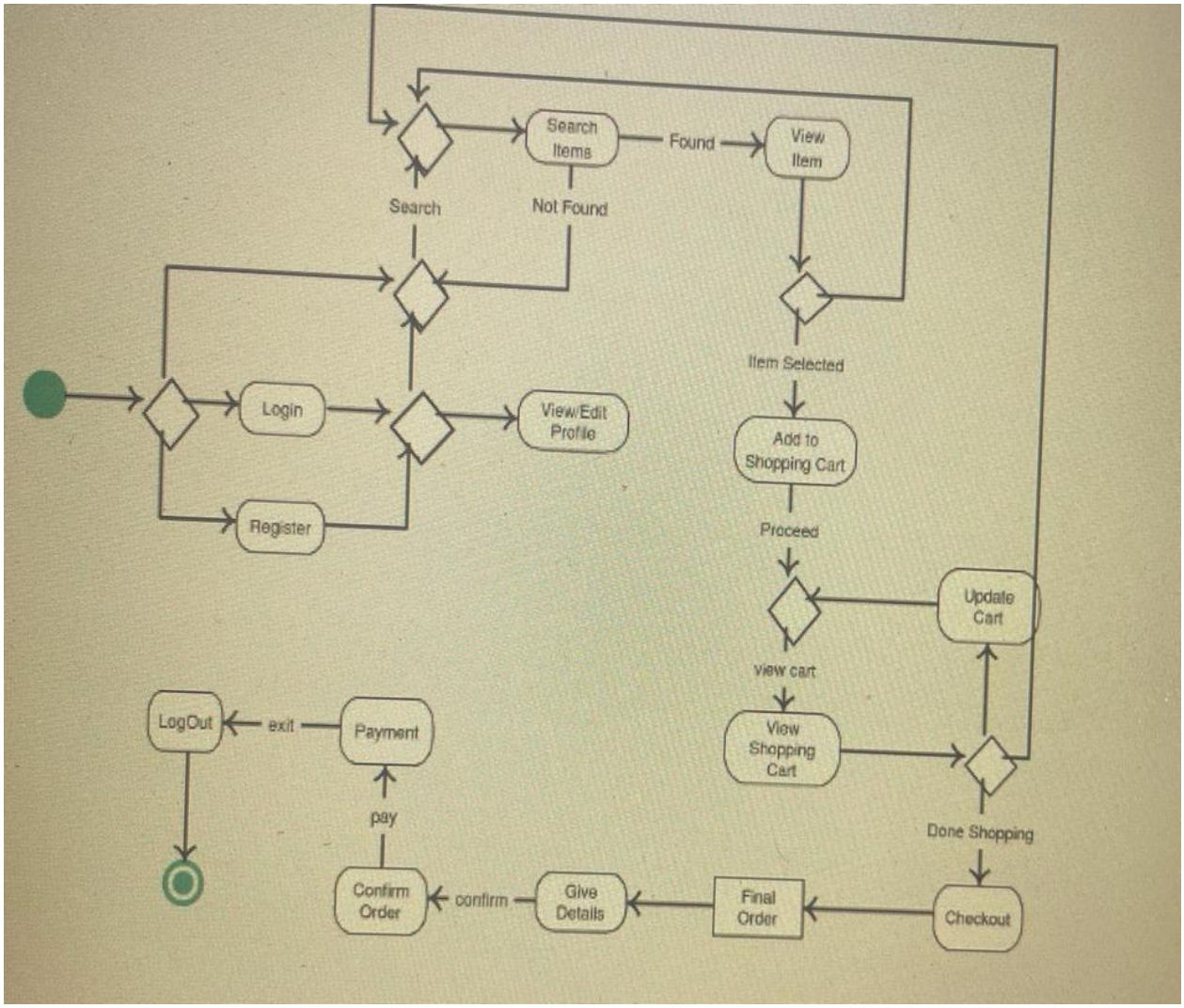
- Payment Gateway
- Order Controller
- Order
- Bill

Messages Between Lifelines:

There are these interactions between system modules represented by lifelines:

- Login
- Choose Product
- Get Product
- View Details
- Add Product to Cart
- Add Product
- Remove Product from Cart
- Remove Product
- Confirm Order (by Shopper)
- Confirm Order (by Order Controller)
- Get Products
- Products
- Create Order
- Pay for Order
- Pay
- Payment Success / Fail
- Create Bill

STATE CHART DIAGRAM



A state diagram is used to represent the condition of the online shopping system or part of the system at finite instances of time. It's a behavioral diagram and it represents the behavior using finite state transitions. State diagrams are also referred to as State machines and State-chart Diagrams. These terms are often used interchangeably. So simply, a state diagram is used to model the dynamic behavior of a class in response to time and changing external stimuli. We can say that each and every class has a state but we don't model every class using State diagrams. We prefer to model the states with three or more states.

They define different states of an object during its lifetime and these states are changed by events. Statechart diagrams are useful to model the reactive systems. Reactive systems can be defined as a system that responds to external or internal events. Statechart diagram describes the flow of control from one state to another state. States are defined as a condition in which an object exists and it changes when some event is triggered. The most important purpose of Statechart diagram is to model lifetime of an object from creation to termination.

In an online shopping system,

1. The user first logs in to his account.

2. After logging in, he has two choices: to search for items to purchase and to check his existing products in the cart.

3. If he chooses the first option, i.e. searching for items to purchase,

1. He will search for the required item in the search box. He will add the item to his shopping cart if the item is found.

2. He may search for more items.

3. If the item is not found, he will again search for the item.

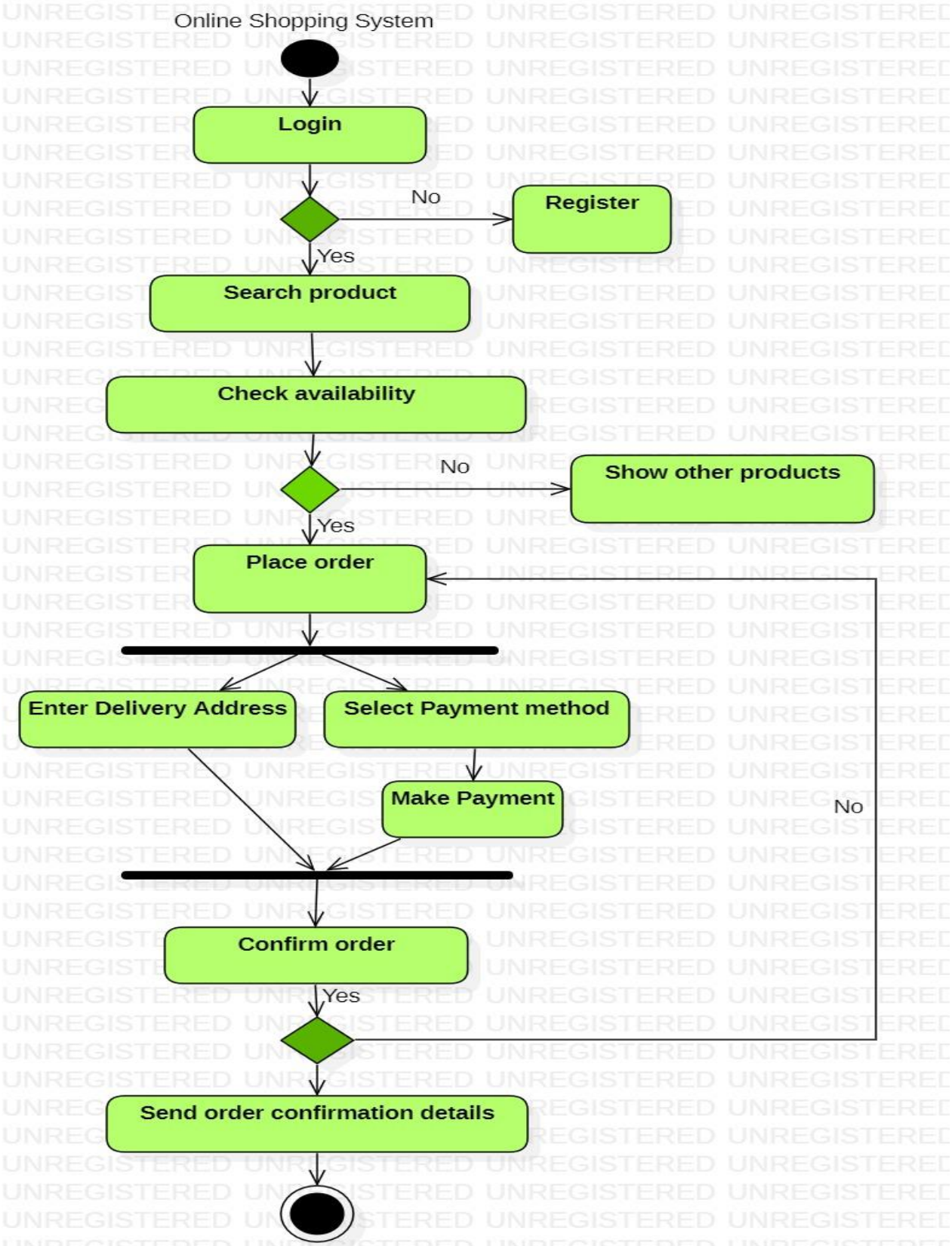
4. After adding the item to the cart, he will see his updated cart and checkout.

4. If he chooses the second option, i.e. checking his existing products in the cart,

1. He will select an item to purchase and will checkout.

5. After placing an order for the item, the user will log out from the system.

ACTIVITY DIAGRAM WITHOUT SWIMLANE

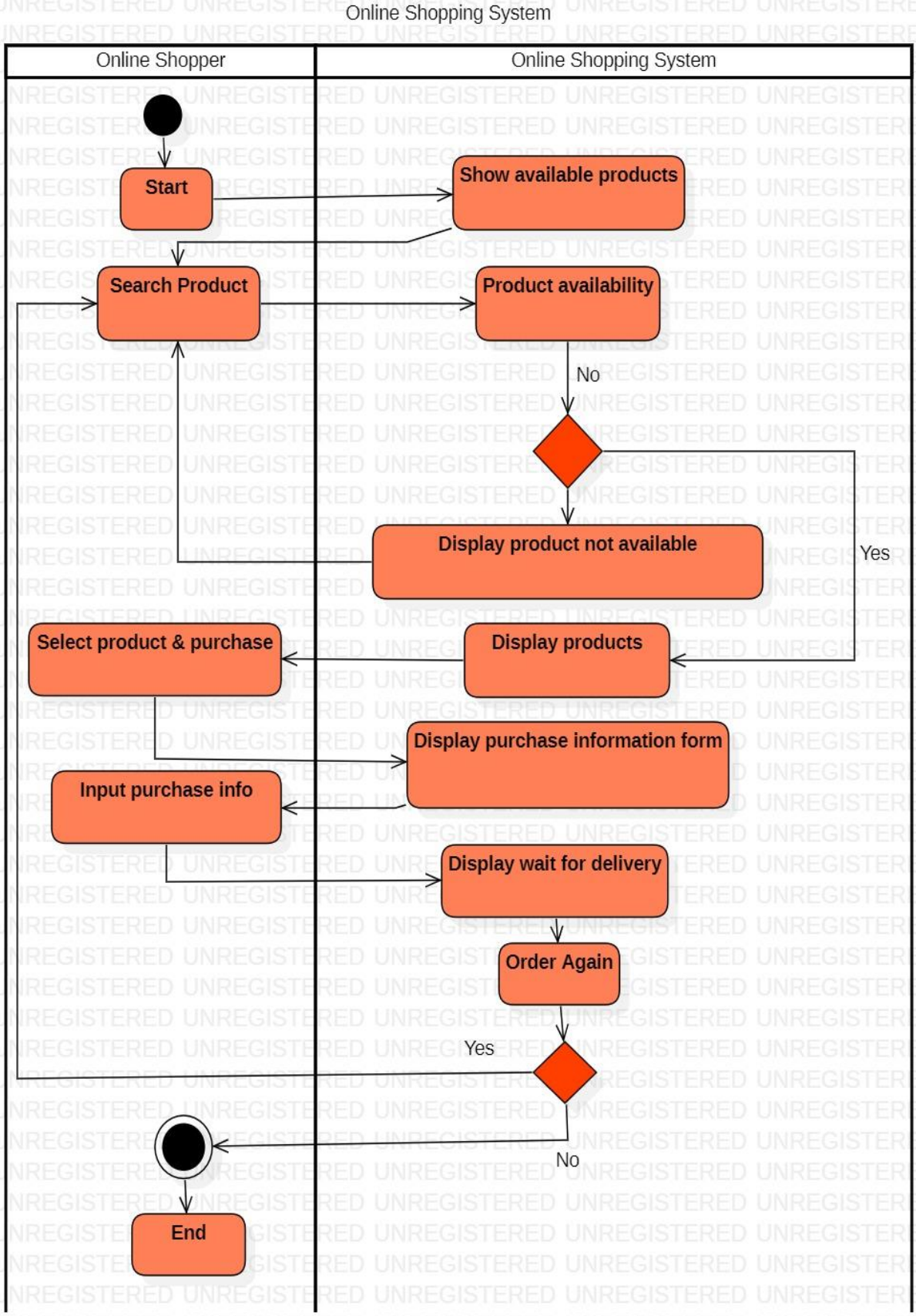


The activity diagram for online shopping system is used to describe flow of activity through a series of actions. Activity diagram is a important diagram to describe the system. The activity described as a action or operation of the system. In User side activity diagram describe all the functionality or operation of users can do on our website. Online customer can browse or search items, view specific item, add it to shopping cart, view and update shopping cart, checkout. User can view shopping cart at any time. Checkout is assumed to include user registration and login.

UML has specified a set of symbols and rules for drawing activity diagrams. Following are the commonly used activity diagram symbols with explanations:

- Start/ Initial Node - Used to represent the starting point or the initial state of an activity
- Activity / Action State - Used to represent the activities of the process
- Action - Used to represent the executable sub-areas of an activity
- Control Flow / Edge - Used to represent the flow of control from one action to the other
- Activity Final Node - Used to mark the end of all control flows within the activity
- Flow Final Node - Used to mark the end of a single control flow
- Fork - Used to represent a flow that may branch into two or more parallel flows
- Merge - Used to represent two inputs that merge into one output

ACTIVITY DIAGRAM WITH SWIMLANE



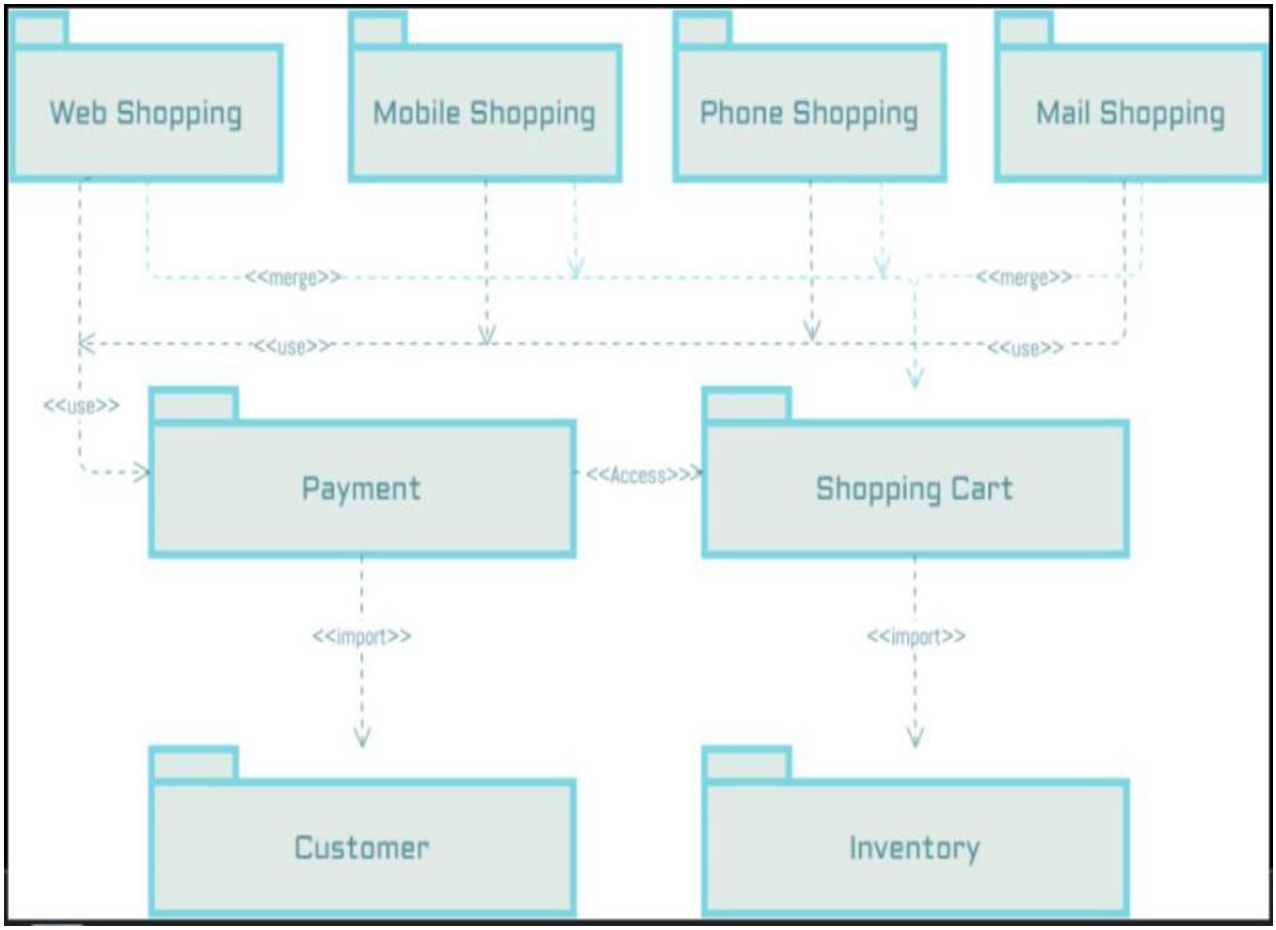
This Swimlane activity diagram for an online shopping system is a UML design diagram that depicts the interaction between the seller, the shoppers, and the system. This will provide you with scenarios of users exchanging activities with the system. Swimlane activity diagrams for online shopping systems contain symbols such as swimlanes, actors, and arrows that will assist you in understanding the workflow of the online shopping management system. The sellers and shoppers are the users of this Online Shopping Management System, and they are essential in creating its activity diagram. It will be much easier for you to build the interactions and activities in the system if you first determine who the system's users are.

In the Swimlane diagram, the activity diagram is divided according to the class responsible for working or performing out these activities. It simply shows the connection and strong communication between these lanes and is used to highlight waste, redundancy, and inefficiency in a process of an activity or program.

Essential key-points of all the above diagrams :

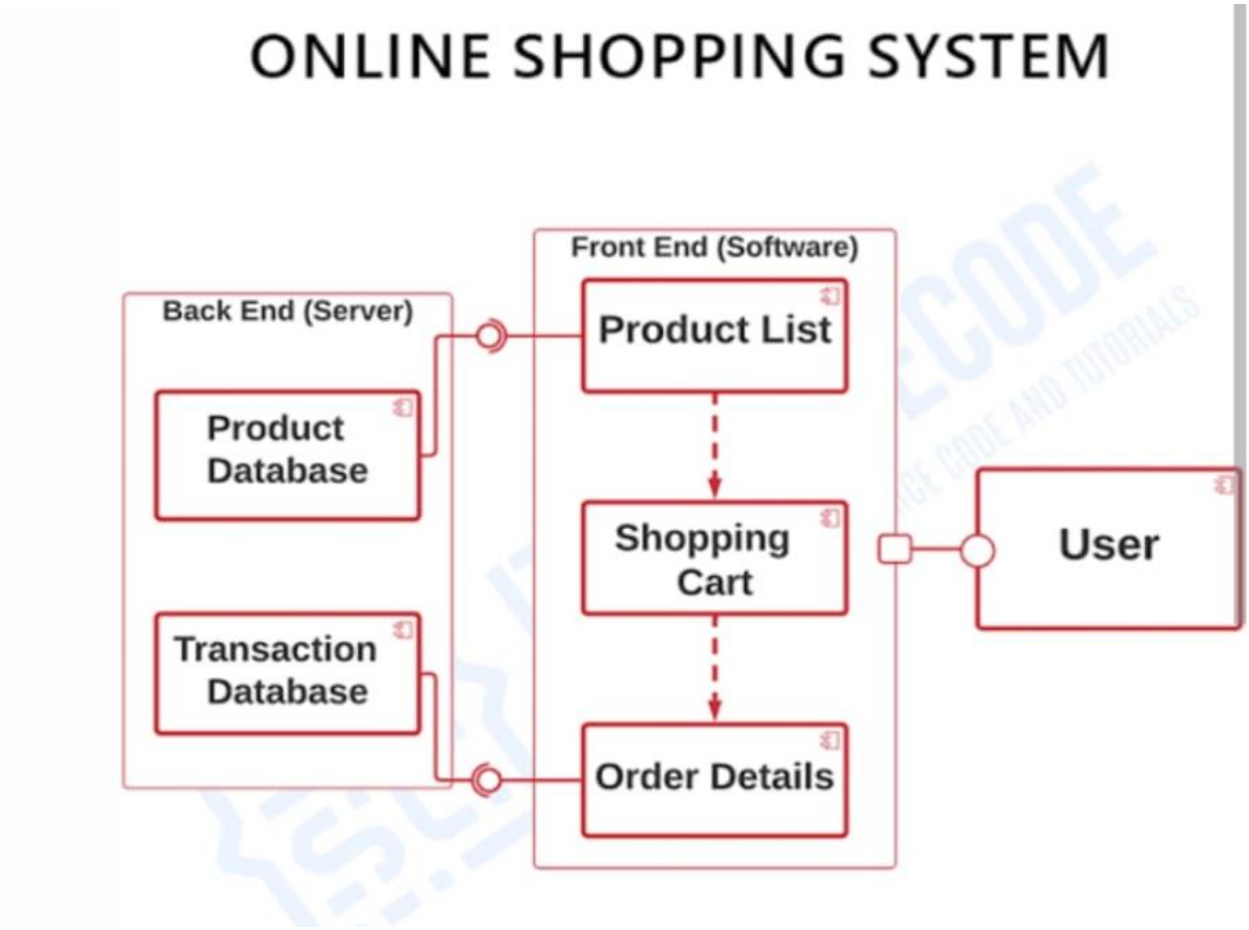
- Fork – It is used to represent the multiple parallel flows.
- Branches – It allow the parallel flow within activities.
- Merge – It brings together or combines together multiple branches.
- Join – It is used to control and synchronize various parallel flows.

PACKAGE DIAGRAM



This Online Shopping Package Diagram is used to organize the components/elements of a high-level system so that packages can be used for large system organizations containing diagrams, documents, and other project goals. The goal is to develop a systematic representation of instructions or procedures. The package diagram follows the package hierarchy. As a result, the package name in the system should not be the same, but classes in different packages may not have the same name. A package can contain any element for each UML, including classes, interfaces, modules, nodes, use cases, diagrams, and other packages.

COMPONENT DIAGRAM



The component diagram of an online shopping system is used to show how the parts of a system work together to make the online shopping operate. A component diagram shows how the software's parts are organized and how they depend on each other. This diagram gives a high-level look at the parts of a system. Components of an online shopping component diagram can be part of software or hardware. They could be a database, a user interface, or something else that helps the online shopping system work.

This component diagram of online shopping system is the illustration of the components of every hardware and software node. This component diagram shows the structure of the online shopping system, which consists of the software components and their interfaces, and how they work together. We can use

component diagrams to show how software systems work at a high level, or we can use them to show how each component works at a lower level, like in a package.

The Online Shopping System UML component diagram explains the sketch of the required software and hardware components and the dependencies between them. These components are labeled to clarify their part in the system's operation. They were represented by symbols that explain their function and role in the overall online shopping operation.

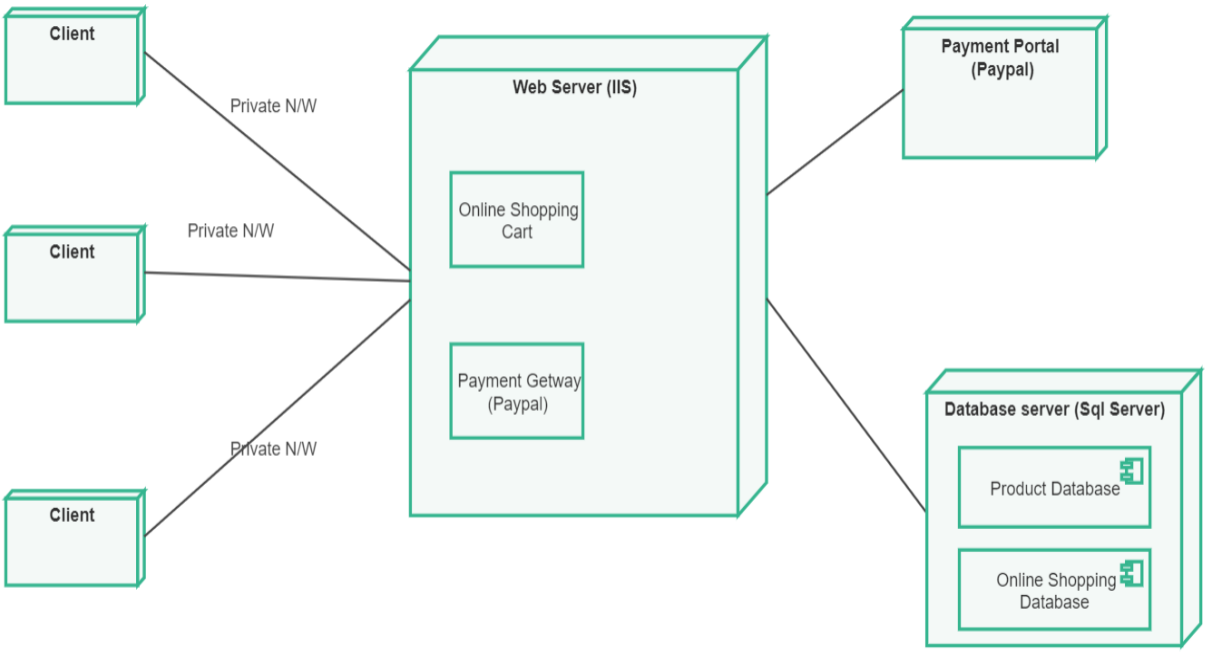
The component diagram of online shopping system has 6 components which are the product database, transaction database, product list, shopping cart, order details, and the user. The component product list is the required interface which is dependent on the provided component Product database. The component transaction database is also dependent to order details.

In object-oriented programming, a senior developer can use the component diagram to group classes together based on their common purpose. This way, the developer and others can see how the online shopping system is progressing at a high level. The UML component diagram shows how an online shopping system will be made up of a set of deployable components, such as dynamic-link library (DLL) files, executable files, or web services. Using well-defined interfaces, these parts communicate with each other and keep their internal details hidden from each other and the outside world.

Therefore, component diagrams show how the online shopping system is put together. During the design phase, software artifacts (classes, interfaces, etc.) of a system are put together into different groups based on how they work together. Now, these groups of things are called parts.

DEPLOYMENT DIAGRAM

Online Shopping System



The Deployment Diagram for Online Shopping System is the manifestation of how the software and hardware are deployed in operating online shopping. It contains the details and design which will serve as a guide for users on how the software work in reality. It consists of nodes such as the software itself, the users' devices, and their connections. The deployment diagram is one of the UML models used to visualize the deployment of the Online Shopping System architecture. It contains elements such as hardware, software, and the middleware that connects them. UML Deployment Diagram presents the system's needs for hardware and software.

Deployment diagrams demonstrate how software and hardware communicate to ensure the appropriate operation of the Online Shopping System.

It properly explains how software interacts with hardware. This also helps figure out which software parts are used by a certain type of hardware.

Online Shopping System uses a UML deployment diagram to show how should the developed software be deployed. It clarifies the communications between links(nodes) which helps the project to work according to the design given to it.

The UML Deployment diagram for Online Shopping System is used to illustrate the system's physical architecture. In UML, deployment diagrams can show you how the software and hardware of online shopping work together and where the processing takes place.

The benefits of a deployment diagram are:

- It aids in the visualization of the various aspects involved.
- Aids in a more accurate description of all the hardware elements used by software components.
- It clarifies the description of the runtime involved in processing nodes.
- Provides hardware specified details for a distributed application.
- Helps in modeling the system's hardware topology.
- It aids in the modeling of inserted or included software.
- Provides more information on the hardware system.
- Reverse engineering is made easier using the UML deployment diagram.

Therefore, it is important to create the deployment diagram in order to clarify the needs of the project before putting it into operation. This will help us avoid unnecessary difficulties that may encounter because of specification deficiency.

UML EXTRACTED CODE

```
package Online Shopping System;
import java.util.*;
public class Admin {
 /**
     * Default constructor
     */
    public Admin() {
    }
    public integer id;
    public string name;
    public string email;
    public void Attribute3;
    public void add_products() {
        // TODO implement here
    }
    public void delete_products() {
        // TODO implement here
    }
    public void modify_products() {
        // TODO implement here
    }
}
```

```
public class Cart {
/**
     * Default constructor
     */
    public Cart() {
}
    public integer id;
    public void Attribute1;
    public integer quantity;
    public float total_price;
}
public class Order status {
 /**
     * Default constructor
     */
    public Order status() {
}
    public void not received;
    public void received;
    public void shipped;
    public void delivered;
    public void cancelled;
}
public class Payment {
```

```
 /**
     * Default constructor
     */
    public Payment() {
}
    public string Cust_name;
    public integer Cust_id;
    public string card_name;
}
public class Products {
 /**
     * Default constructor
     */
    public Products() {
    }
    public integer product_id;
    public string name;
    public float price;
}
public class Customer {
/**
     * Default constructor
     */
    public Customer() {
```

```
    }
    public integer id;
    public string name;
    public integer phone_no;
    public string email;
    public string address;
    public void login() {
        // TODO implement here
    }
    public void searchproduct() {
        // TODO implement here
    }
    public void add_items() {
        // TODO implement here
    }
    public void make_payment() {
        // TODO implement here
    }
    public void logout() {
        // TODO implement here
    }
}
public class Guest user {
/**
```

```
     * Default constructor
     */
    public Guest user() {
    }
    public void v;
    public void view_products() {
        // TODO implement here
    }
}
```

CONCLUSION

Therefore, UML Diagrams for Online Shopping System have been used to represent the online shopping system as well as its primary users, roles, activities, artifacts, or classes. The UML Diagrams are created to easily understand, update, maintain, and document online shopping system information. UML diagrams for online shopping system were used to visualize the project. It can be done before the development begins or to document its progress once it is completed. However, Online Shopping System UML Diagrams can be used in any sector, not only in software engineering. Its overall objective is to help teams or developers visualize what a project is or how it will work.

REFERENCES

a. https://itsourcecode.com/uml/online-shopping-complete-uml-diagrams/

b. https://meeraacademy.com/use-case-diagram-for-online-shopping/

c. https://www.edrawmax.com/templates/1007460/

d. https://www.uml-diagrams.org/examples/online-shopping-uml-component-diagram-example.html?context=cmp-examples