DOKUZ EYLÜL UNIVERSITY

ENGINEERING FACULTY

DEPARTMENT OF COMPUTER ENGINEERING

CME 2210

Object Oriented Analysis and Design

DEU-SHOP

by

Berkay Coşkuner-2016510015

Berk Adalı-2016510001

Elif Karataş-2016510040

CHAPTER ONE

INTRODUCTION

The purpose of this document is to build an online system to manage shopping and customers to publishing and selling people's undressing or unused clothes and products.

The aim of the online shopping management system is to ease shopping and build an online platform to sender and receiver customers in all cities.

The system works with selling points and buying points with functions. We aim at simple, cheaper and faster shopping life in this project. System has 3 perspectives. Customer, manager, seller. The user can choose one of them and then can start the program.In common, they have login system. The user must enter a correct username-password pair. After a succesfull login you can continue the system with selected perspective.

This program provides to buying and selling in easier way to all of the people. Main purpose of this system is cheaper shopping and 7/24 ordering.

Thus, this network can envelop all around the world.

CHAPTER TWO

REQUIREMENTS

3.1 External Interfaces User Interfaces:

- No requirements.

Hardware Interfaces:

- Windows
- An IDE which support JDK 1.8 and upper - NET.Framework 4.7

Software Interfaces: Following are the software used for DEU-SHOP online shopping system.

Operating System: Windows operating system selected for ease of use and userfriendliness.

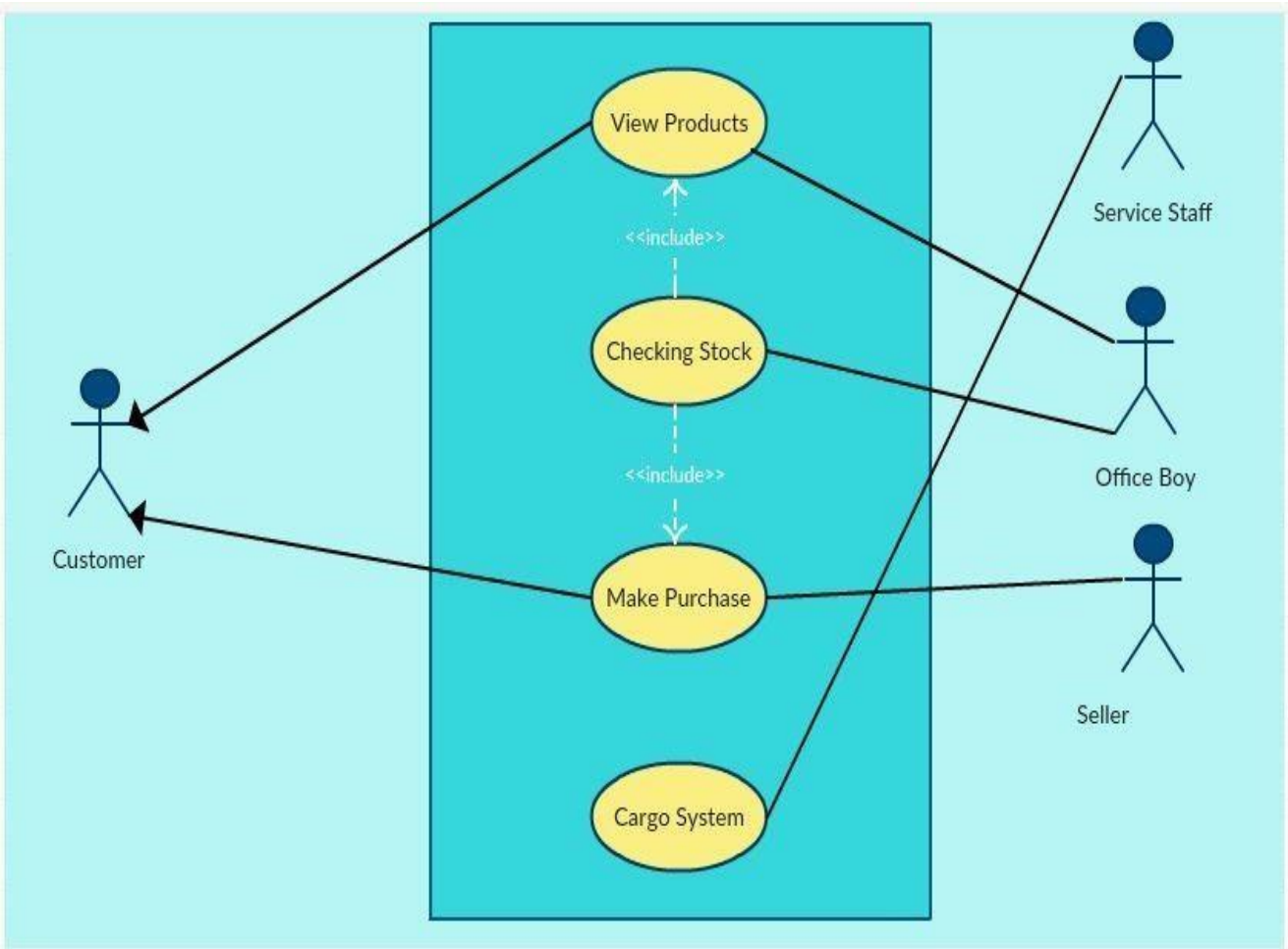
Eclipse: Eclipse IDE selected for speed and performance power.

Java: Java selected for its rich libraries and for OOP support.

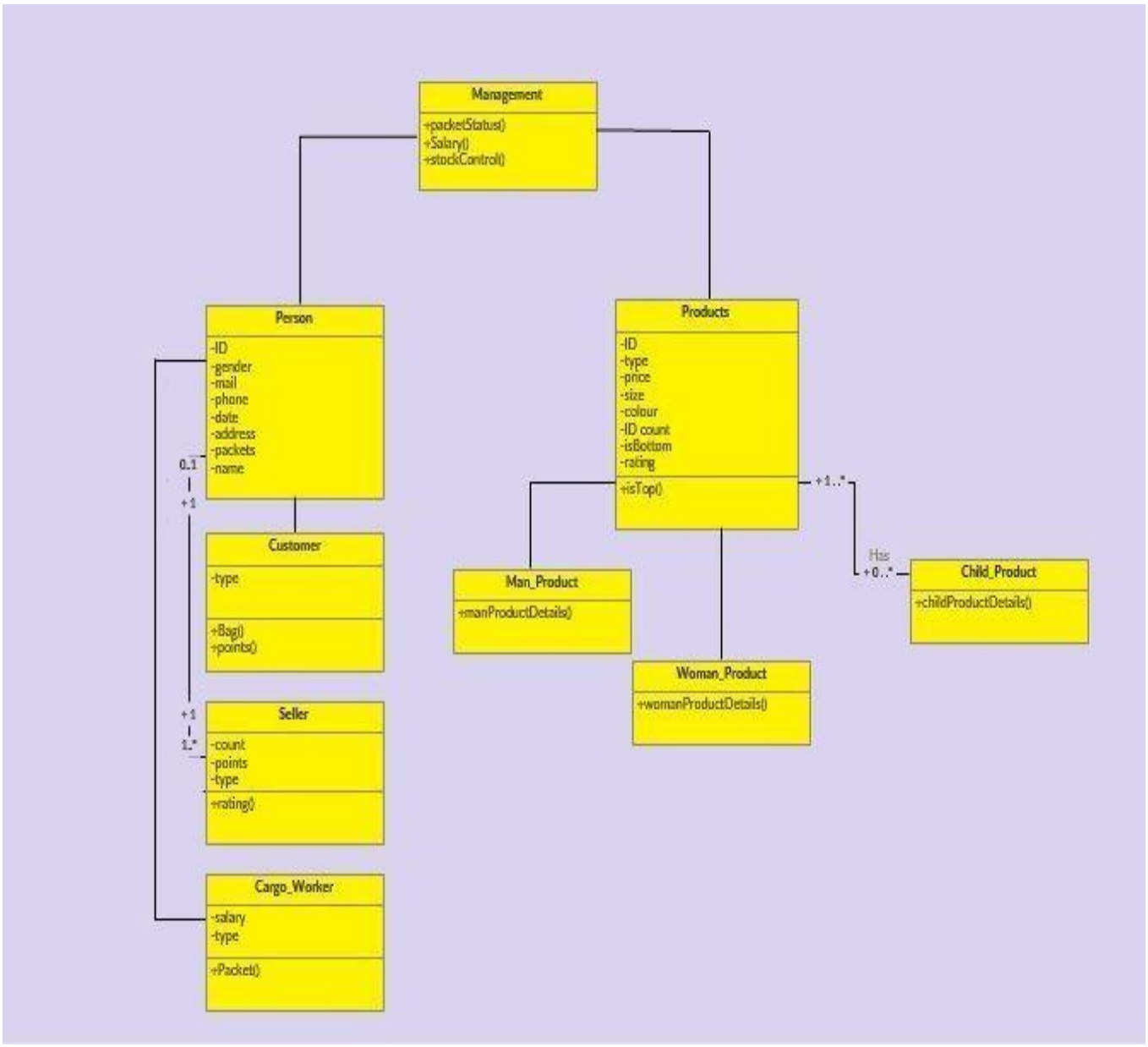
CHAPTER THREE

UML DIAGRAMS

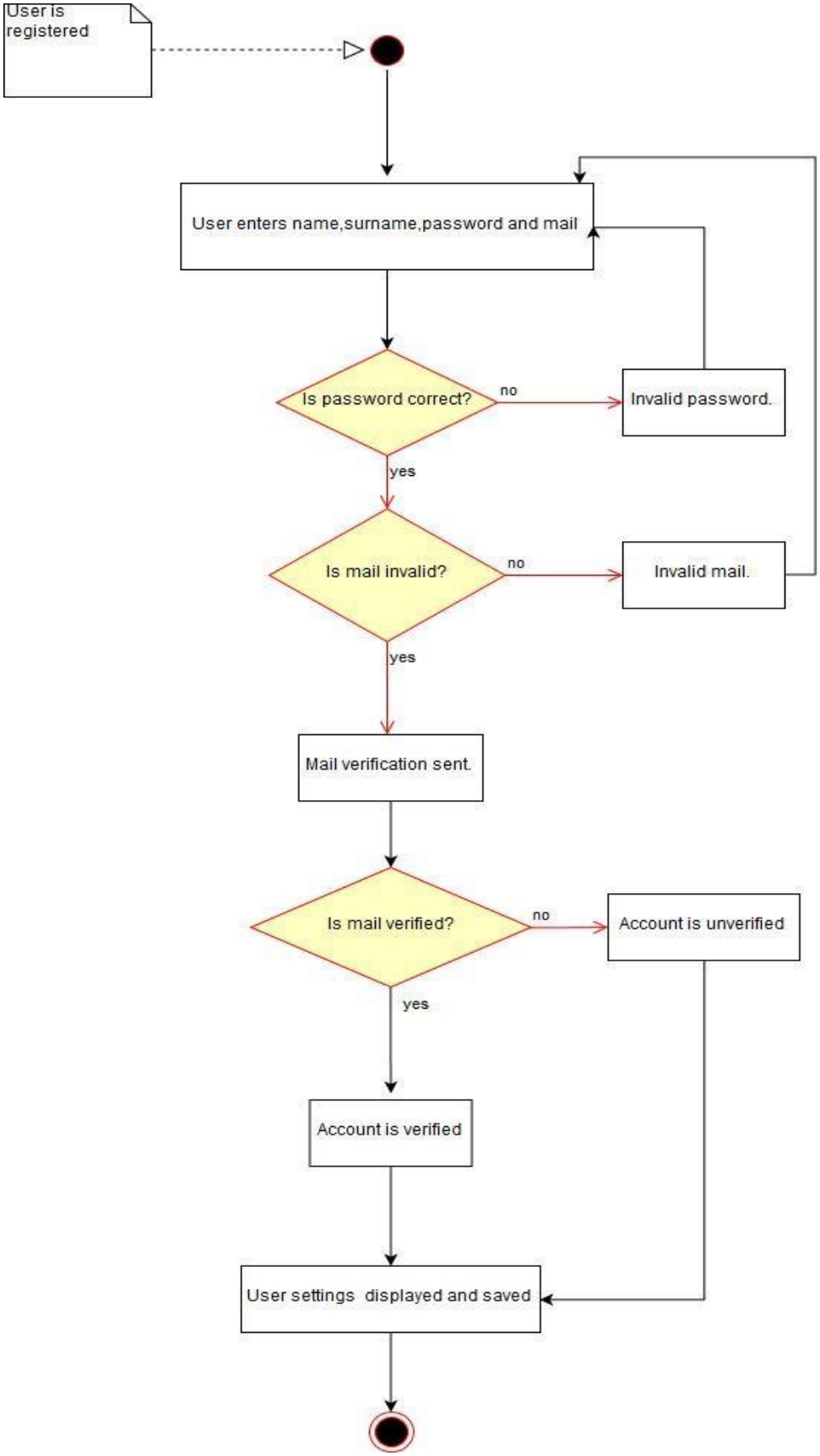
1.Use Case Diagram



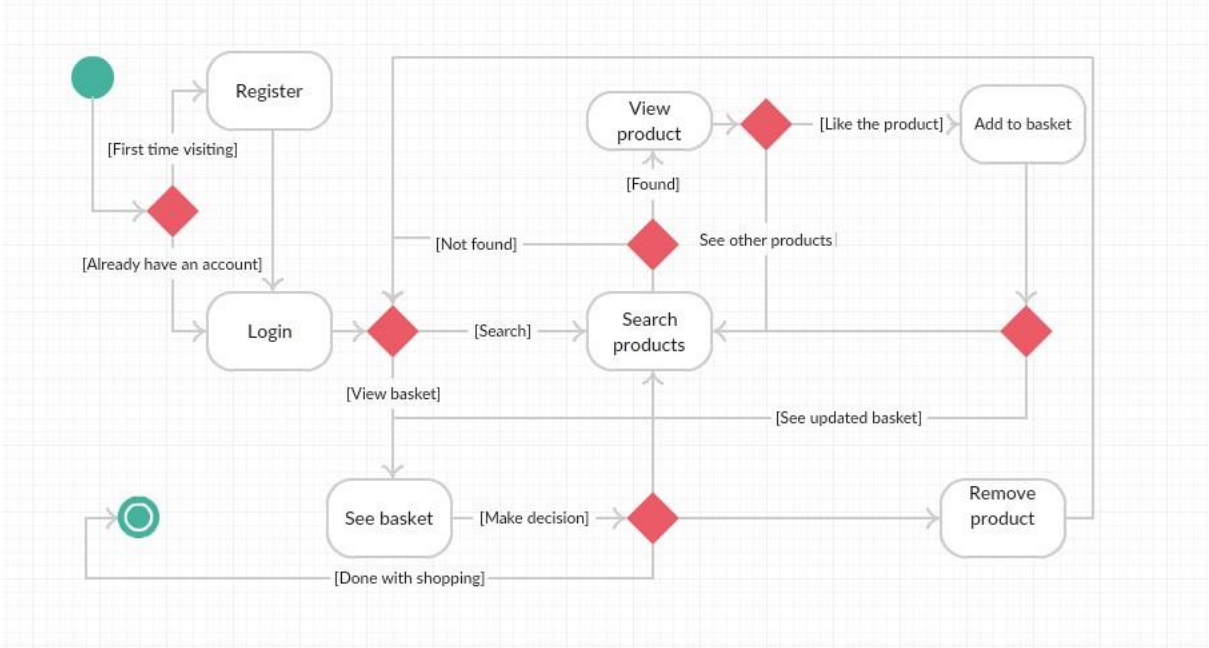
2.Class Diagram



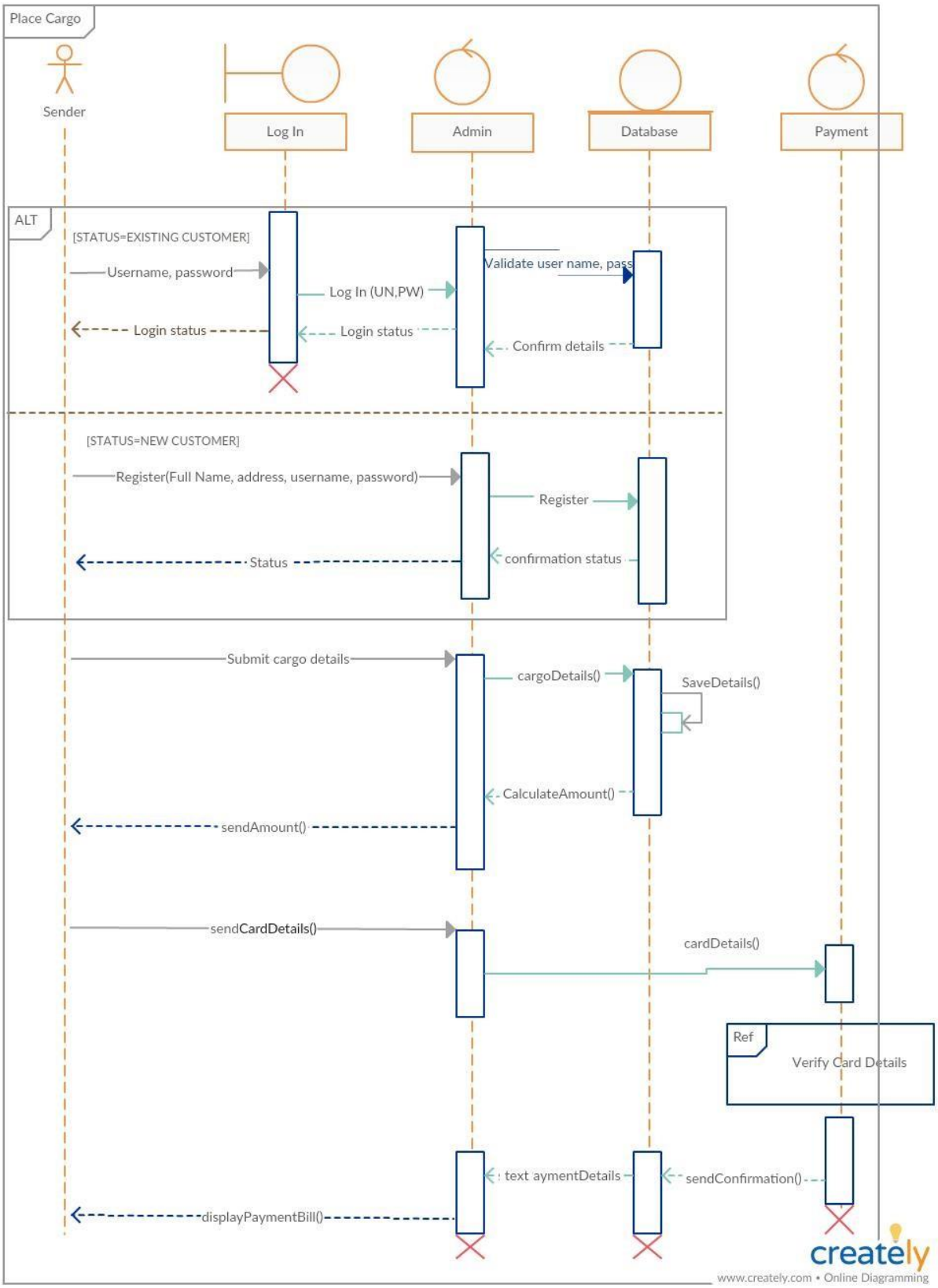
3.Activity Diagram Login System



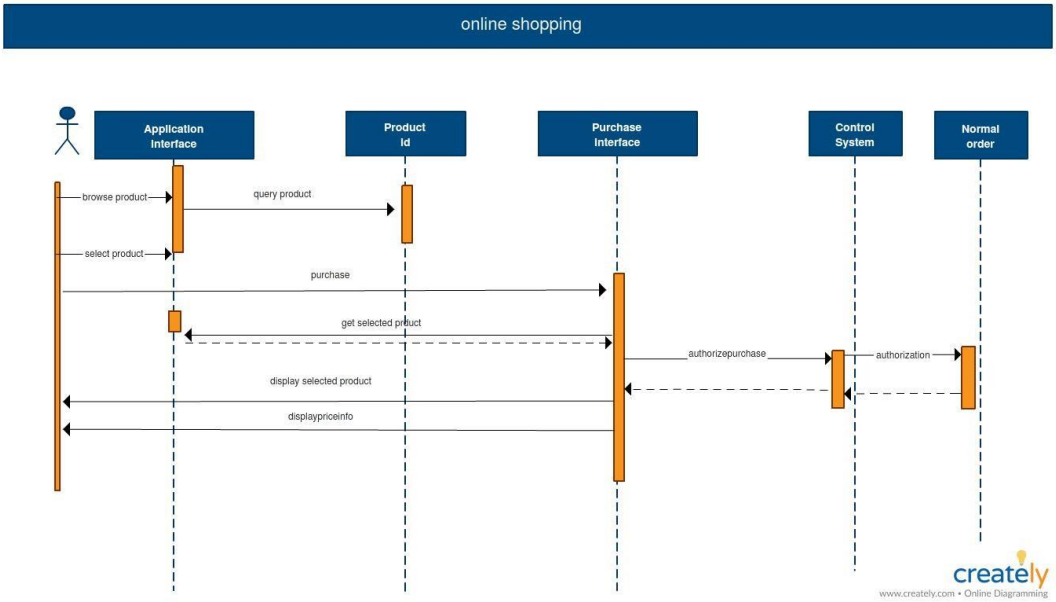
Shopping System



4.Sequence Diagram Cargo System

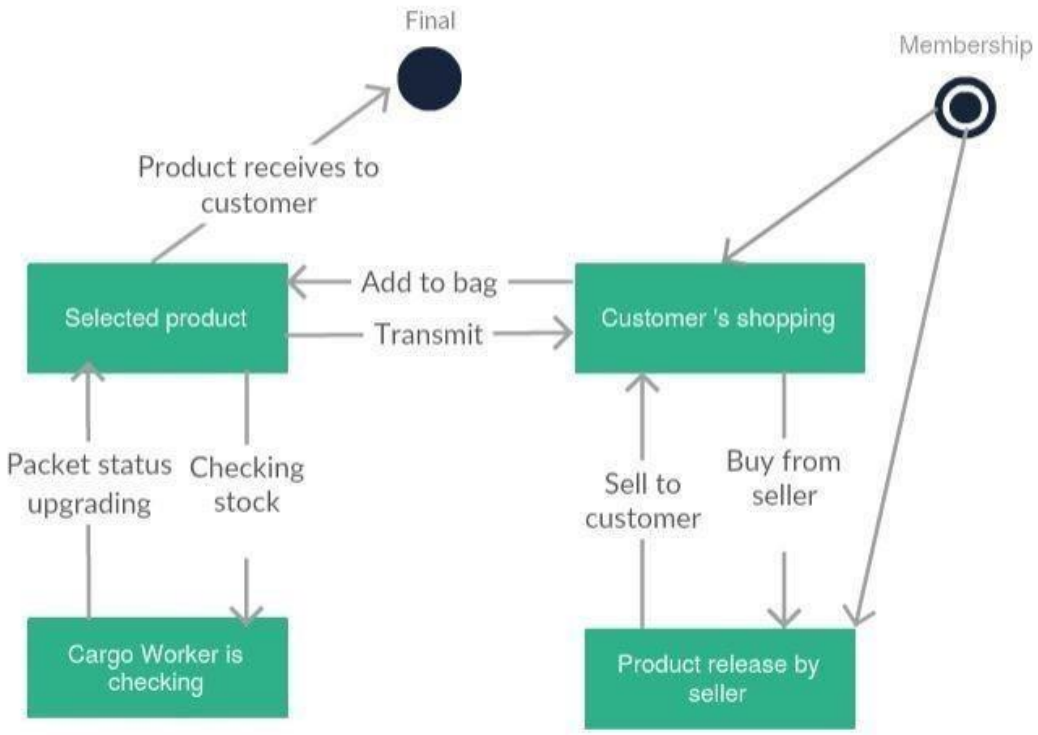


ShoppingSystem

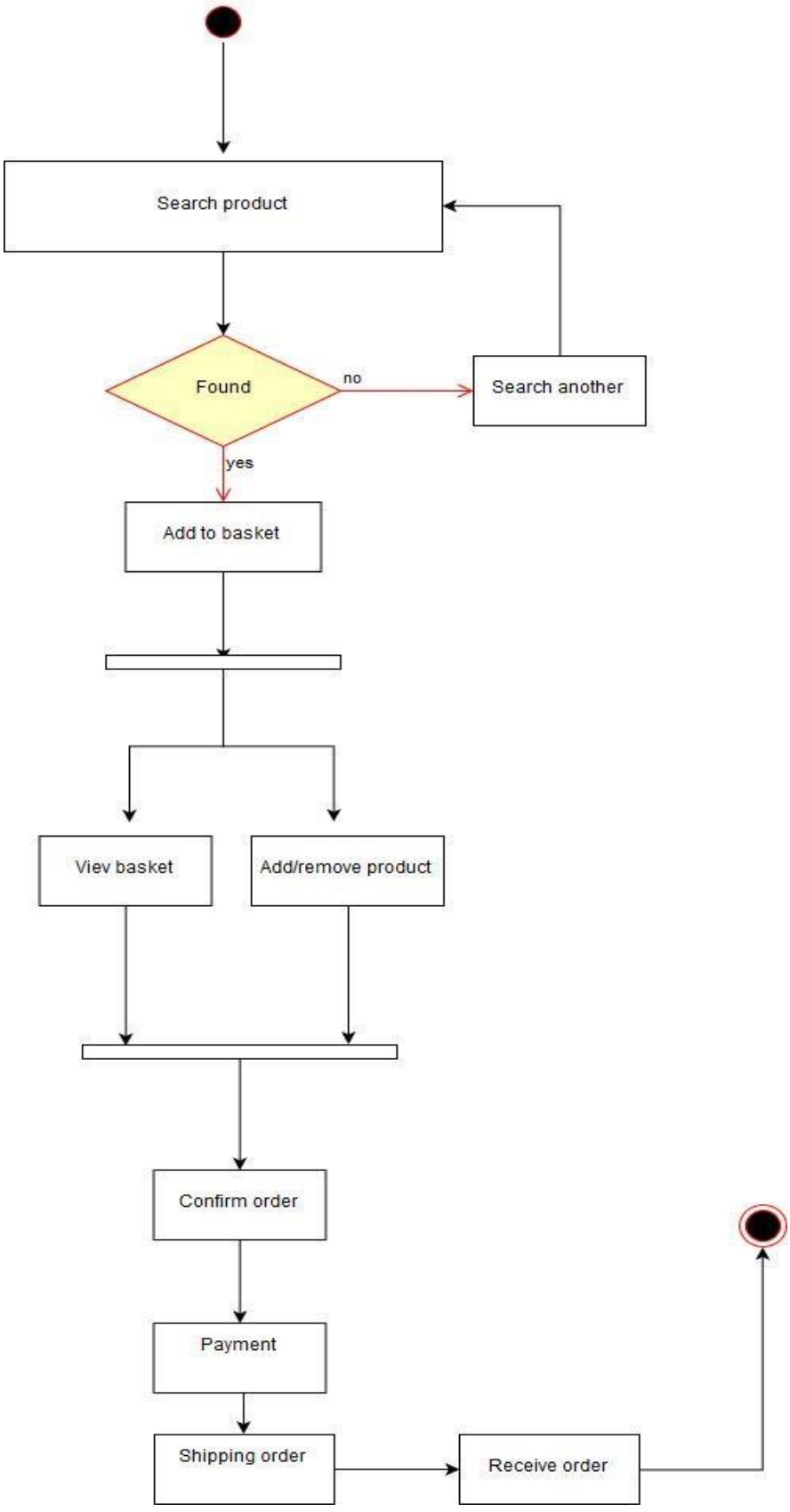


5.State Diagram

Shipping System



Shopping System



CHAPTER FIVE
IMPLEMENTATION

In the project, there is a **"IProduct"** Interface. This interface has six methods for abstract class which is Product.
First method for tracking the cargo with "Customer" which is identified Customer, "Seller" which is identified Seller, "D1" and "D2" which is identified Date. For all of the workers, If workers' current packets are less than ten, a new packet will be created. After the creating, packet status will be "InCargo". This means "enqueue()" new packet to seller's "orders" that created with Queue structure. At the same time, Same packet will be added to workers' "distribution_order" which is created in DEUSHOP Class with Queue structure and added to "storage" that created with Stack structure. Following that, seller "reliability" which is identified with integer in Seller Class will be increased. Second method is for the selling products and removing these products with "pop()" which did selled from seller to customers. This basket is created with Stack structure and products are removed from basket with "pop()" operation if product ID which is in the basket same with product id which is selled. After that, Product "selling_count" will be increased for use in "Rating" method and "type_points" will be increased for "setType" method which is returned String type and created in Customer Class. Third method returns "rating" stars that identified with String. This method has if controls with "selling_count".
Fourth method is "makeDiscount" that returns "newPrice" which is identified with integer. This method provides sale for all products with "amount" which is identified with double.
Fifth method is "addToBasket" for add products to Customers' basket with "push()" operation which is created with Stack structure.
Except that the "IProduct" Interface, there are different methods except getter/setter and toString().
In **Seller Class** , there are three methods for "toBeSold" structure which is created with List. In this list, there are products for selling.
First method is "listToBeSoldItems" that list the product items with their index.
Second method is "SearchInToBeSold" that searching the product items with using their ID for the searching product.
Third method is "deleteInToBeSold" that deletes product index the product with using product ID.
In **Customer Class** , there is a method that returns String "type" with using "type_points" in if controls.
In **CargoWorker Class** , there are two methods for distribution order that is created with Queue structure.
First method is "addDistribution_order". Packets which are selled, adding to distribution_order if "currentPacketCount" less than ten. "currentPacketCount" is integer that holds the numbers that cargo workers' packets for that transport.
23 Second method is "deleteDistribution_order". If packet receive to customer, packet will be deleted from cargo worker's distribution order with using packet ID. Therefore, "succesfull_orders" counter which is identified with integer will be increased. In **Address Class** , there is a method for controlling invalid dates with using if controls. This method is named with "controllingDate".
In **DEUSHOP** Class , there are List and Search methods for "persons" array which identified with Person, "products" which is created with List structure, "storage" which is created with Stack structure, "DEC" which is identified with DECargo. In **DECargo**

Class , there are two methods for “workers” that created with Array structure that identified with CargoWorker.
First method of these, list all of the workers.
Second method is search workers with using ID.
Except these methods there are two methods for packets and a method for “storage” which is created with Stack structure.
“ListStorage” method lists all of the packets in cargo system.
“SearchPacket” method searches packet with using packet ID.
“DeletePacket” method deletes packet with using packet ID.
These packet methods works with push() and pop() operations.