VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

Object Oriented Analysis and Design

Submitted by

MOHAMMED IBRAHIM RAHIL S (1BM19CS090)

in partial fulfillment for the award of the degree of
BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

B.M.S. COLLEGE OF ENGINEERING
(Autonomous Institution under VTU)
BENGALURU-560019
April-2022 to July-2022

B. M. S. College of Engineering,
Bull Temple Road, Bangalore 560019
(Affiliated To Visvesvaraya Technological University, Belgaum)
Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the Lab work entitled "**Object Oriented Analysis and Design**" carried out by **Mohammed Ibrahim Rahil S (1BM19CS090),** who is a bonafide student of **B. M. S. College of Engineering.** It is in partial fulfillment for the award of **Bachelor of Engineering in Computer Science and Engineering** of the Visvesvaraya Technological University, Belgaum during the academic year 2021-2022. The Lab report has been approved as it satisfies the academic requirements in respect of an **Object Oriented Analysis and Design - (20CS6PCOMD)** work prescribed for the said degree.

Prof. Shyamala G
Associate Professor
Department of CSE
BMSCE, Bengaluru

Dr. Jyothi S Nayak
Professor and Head
Department of CSE
BMSCE, Bengaluru

`

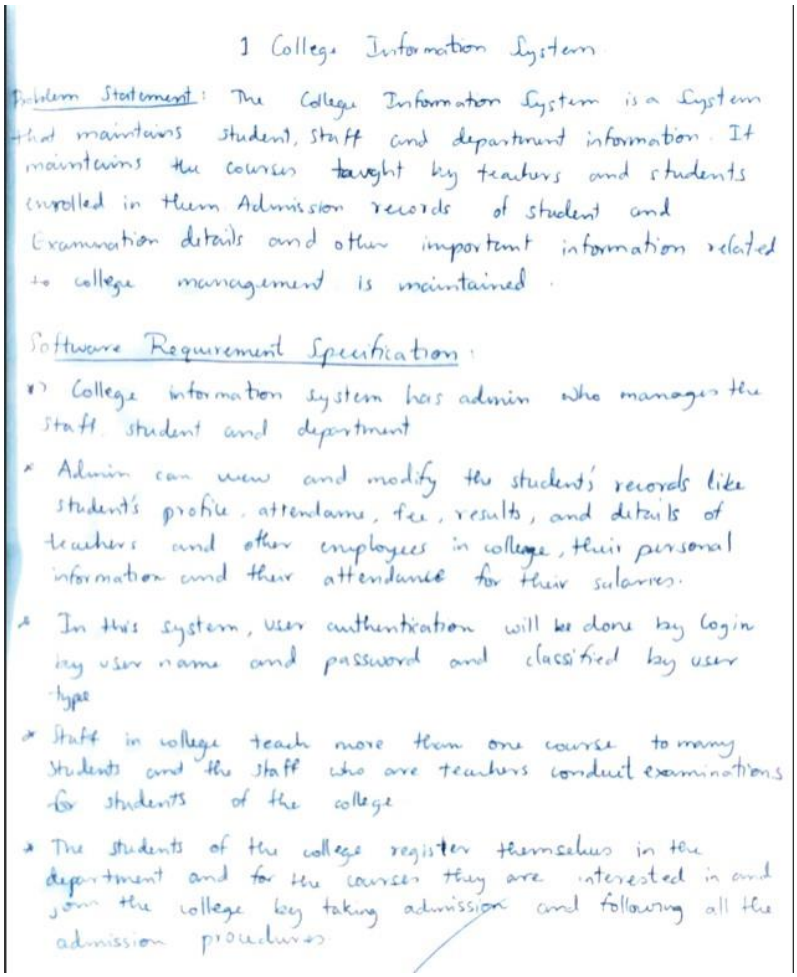
Index Sheet

Course Outcome

CO4	Ability to conduct practical experiment to solve a given problem using Unified Modeling language.

1. College Information System -

a) SRS:



1 College Information System

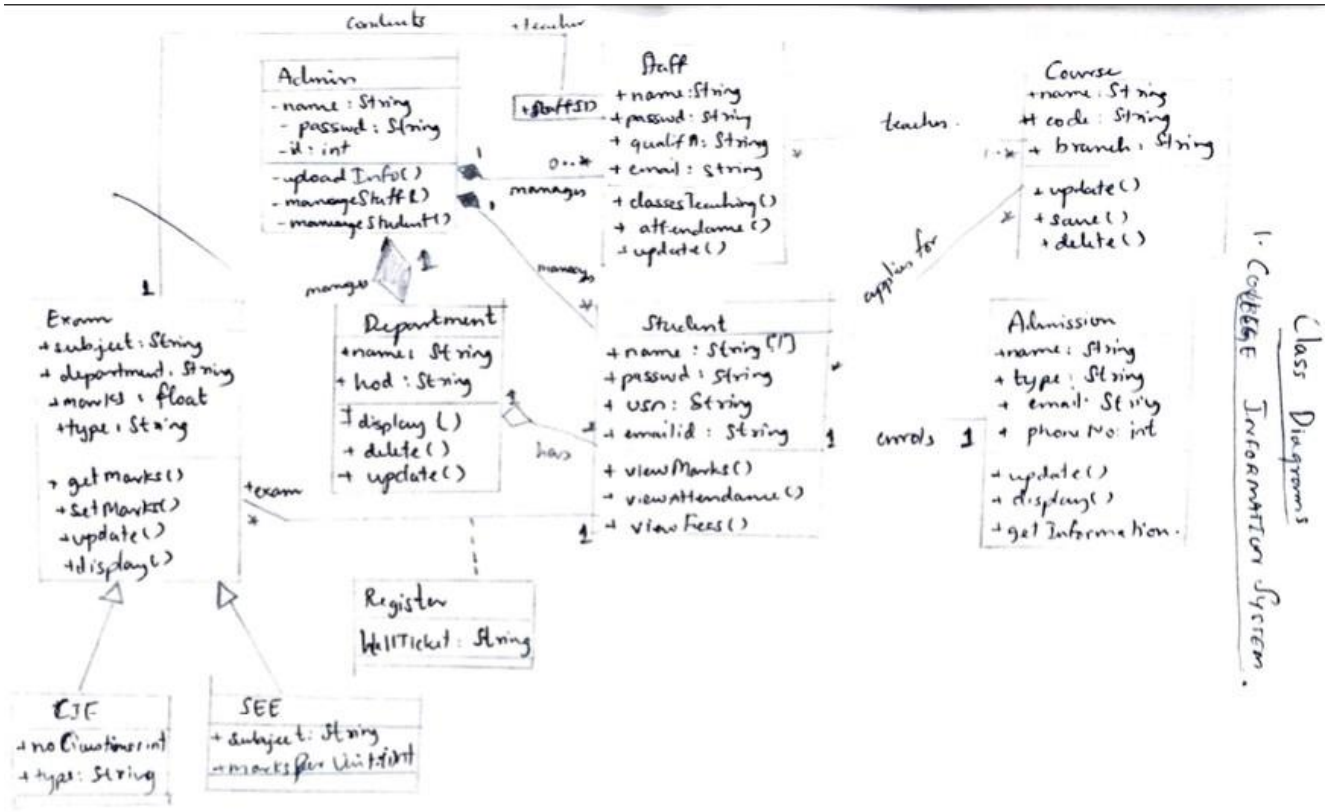
Problem Statement: The College Information System is a System that maintains student, staff and department information. It maintains the courses taught by teachers and students enrolled in them Admission records of student and Examination details and other important information related to college management is maintained.

Software Requirement Specification:

* College information system has admin who manages the staff, student and department
* Admin can view and modify the student's records like student's profile, attendance, fee, results, and details of teachers and other employees in college, their personal information and their attendance for their salaries.
* In this system, user authentication will be done by login by user name and password and classified by user type
* Staff in college teach more than one course to many students and the staff who are teachers conduct examinations for students of the college
* The students of the college register themselves in the department and for the courses they are interested in and join the college by taking admission and following all the admission procedures

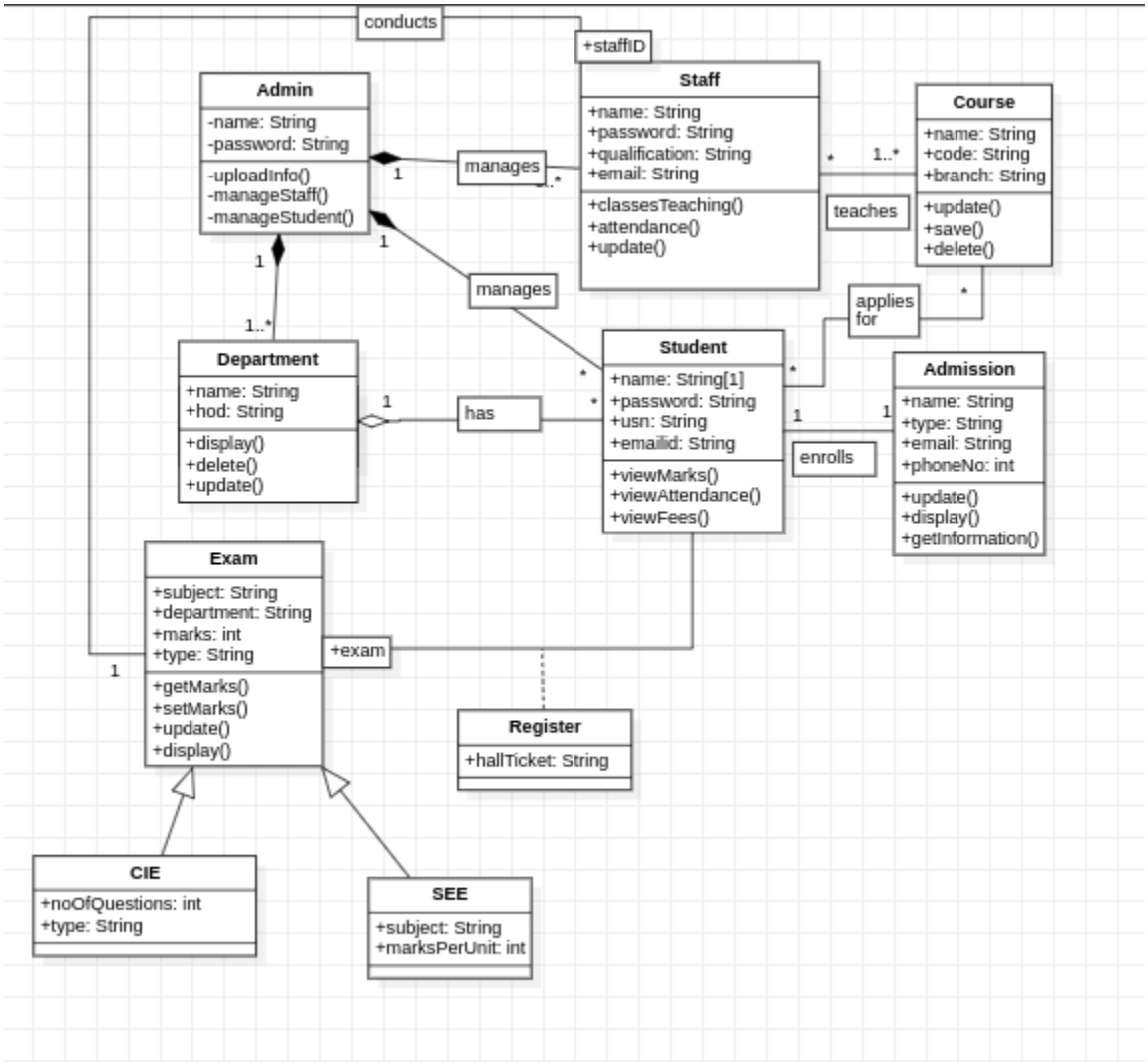
* There are different types of examinations conducted by the college for the students, Internals and semester end examination are 2 of them.
* Every course has a name and its unique name. Every course has different subjects and every subject has its own unique name
* Department will provide the details about departments within a college with their name and every department have its Department name.

b) Advance Class Diagram:

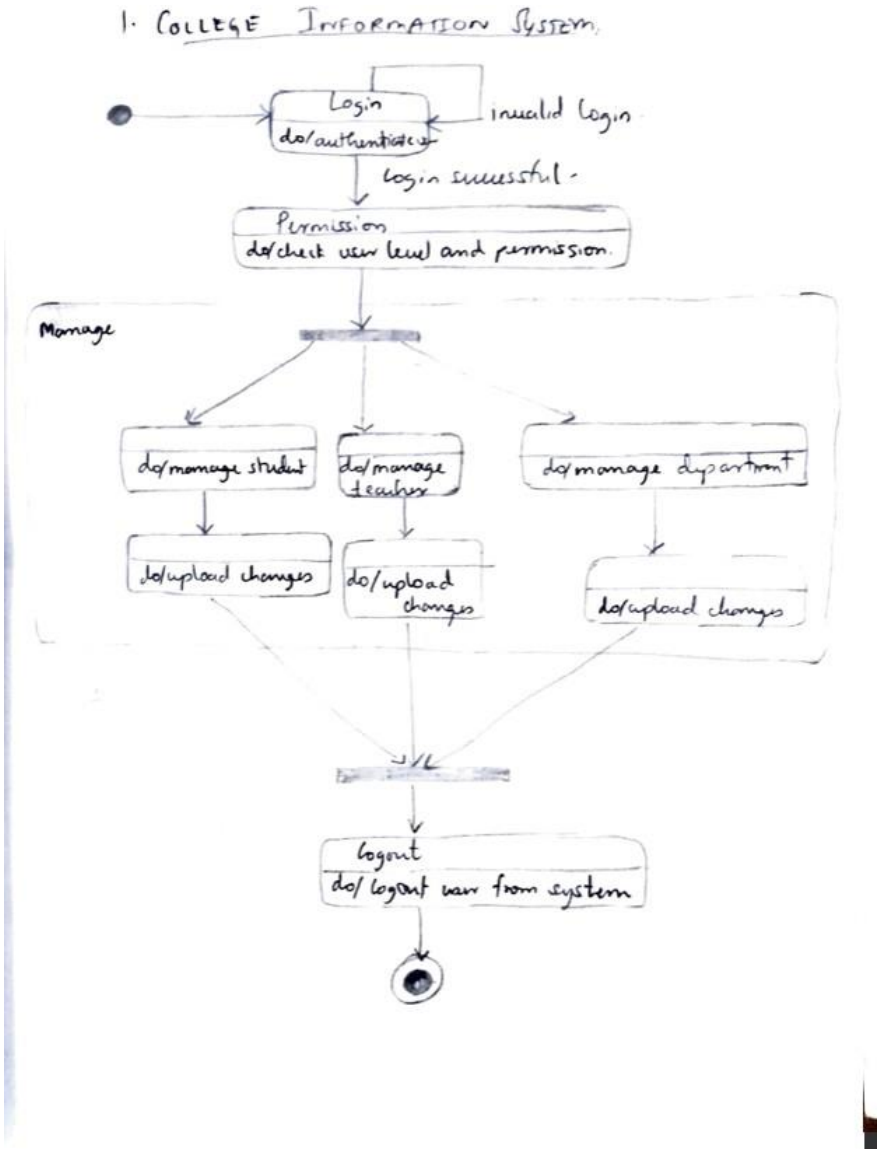


Justification:

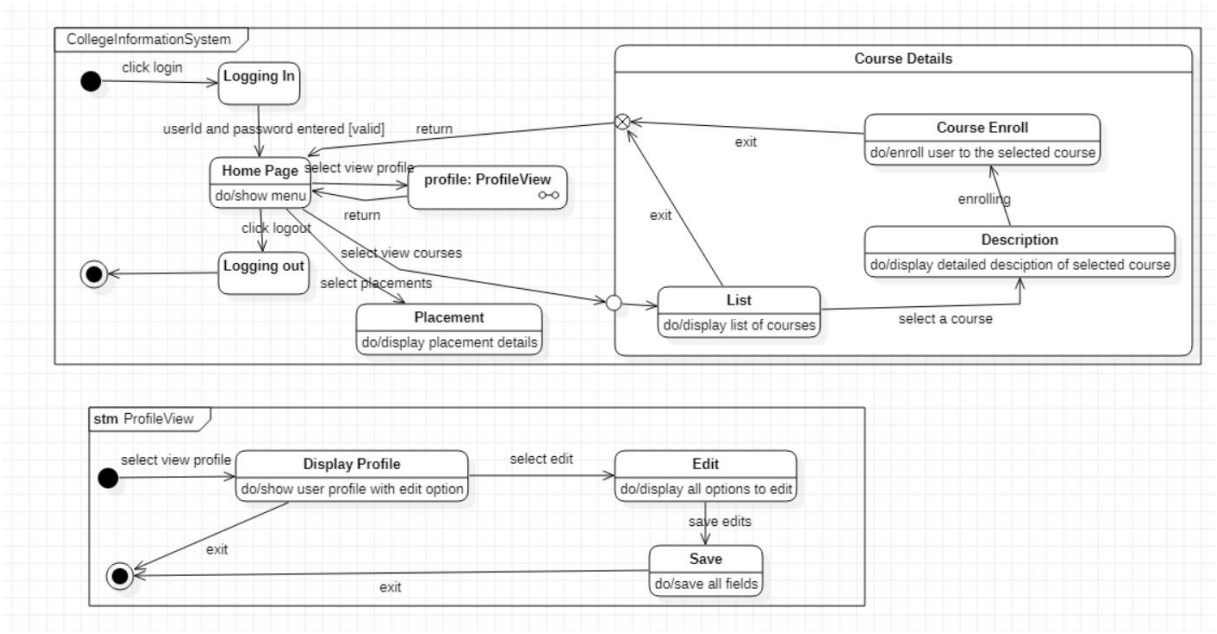
Under Admin, department, student and staff are the components which cannot exist without admin class, hence it is a composition. The courses are associated with the staff as they teach the course. The staff also conduct exams, which have 2 sub classes related by inheritance. Students are associated with class, exam and staff.


conducts
+staffID
Admin
-name: String
-password: String
-uploadInfo()
-manageStaff()
-manageStudent()
manages
Staff
+name: String
+password: String
+qualification: String
+email: String
+classesTeaching()
+attendance()
+update()
teaches
Course
+name: String
+code: String
+branch: String
+update()
+save()
+delete()
manages
applies for
Department
+name: String
+hod: String
+display()
+delete()
+update()
has
Student
+name: String[1]
+password: String
+usn: String
+emailid: String
+viewMarks()
+viewAttendance()
+viewFees()
enrolls
Admission
+name: String
+type: String
+email: String
+phoneNo: int
+update()
+display()
+getInformation()
Exam
+subject: String
+department: String
+marks: int
+type: String
+getMarks()
+setMarks()
+update()
+display()
+exam
Register
+hallTicket: String
CIE
+noOfQuestions: int
+type: String
SEE
+subject: String
+marksPerUnit: int

c) Advance State Diagram:

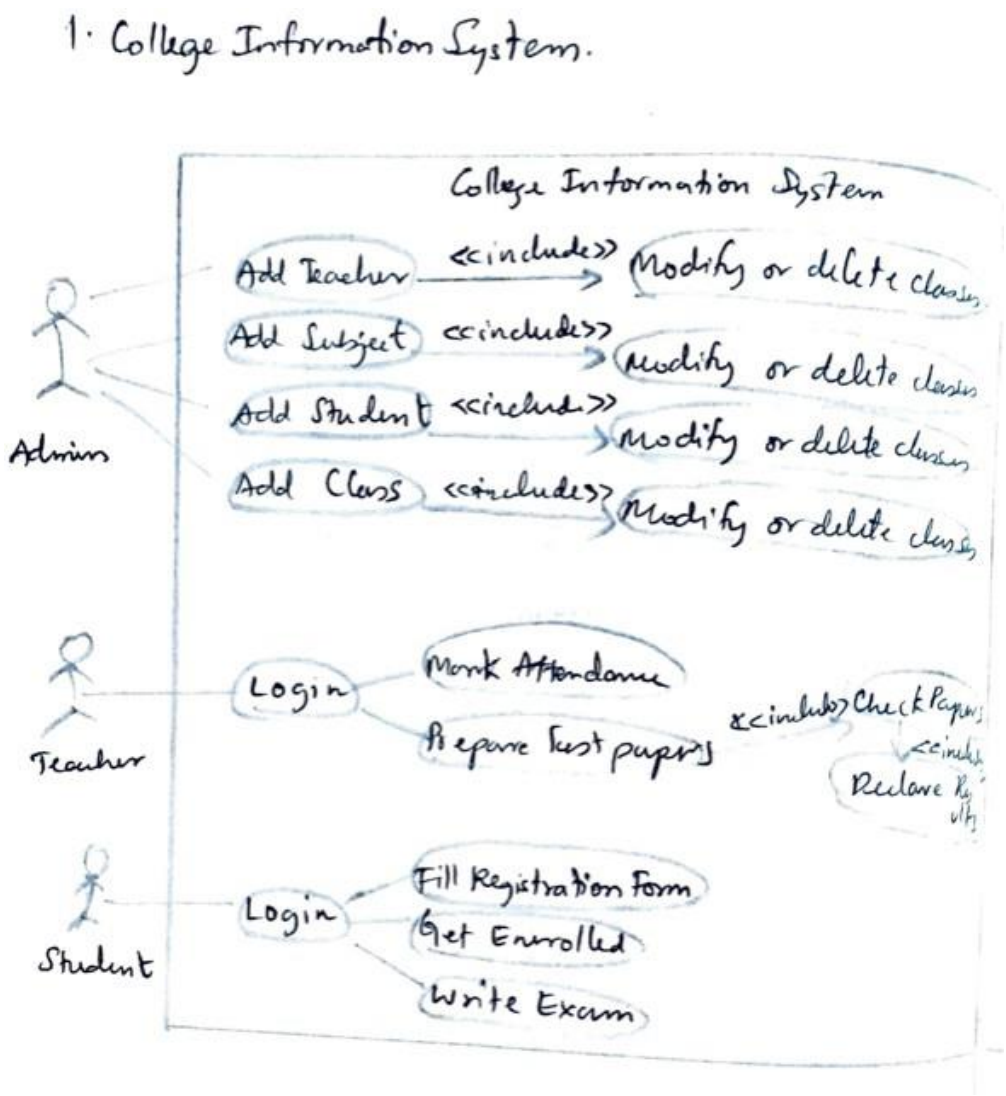


Justification: The given state diagram explains the various states present in the case and gives a detailed description of course details and view profile in the form of a sub state machine and composite machine. All the respective details of transitions are mentioned

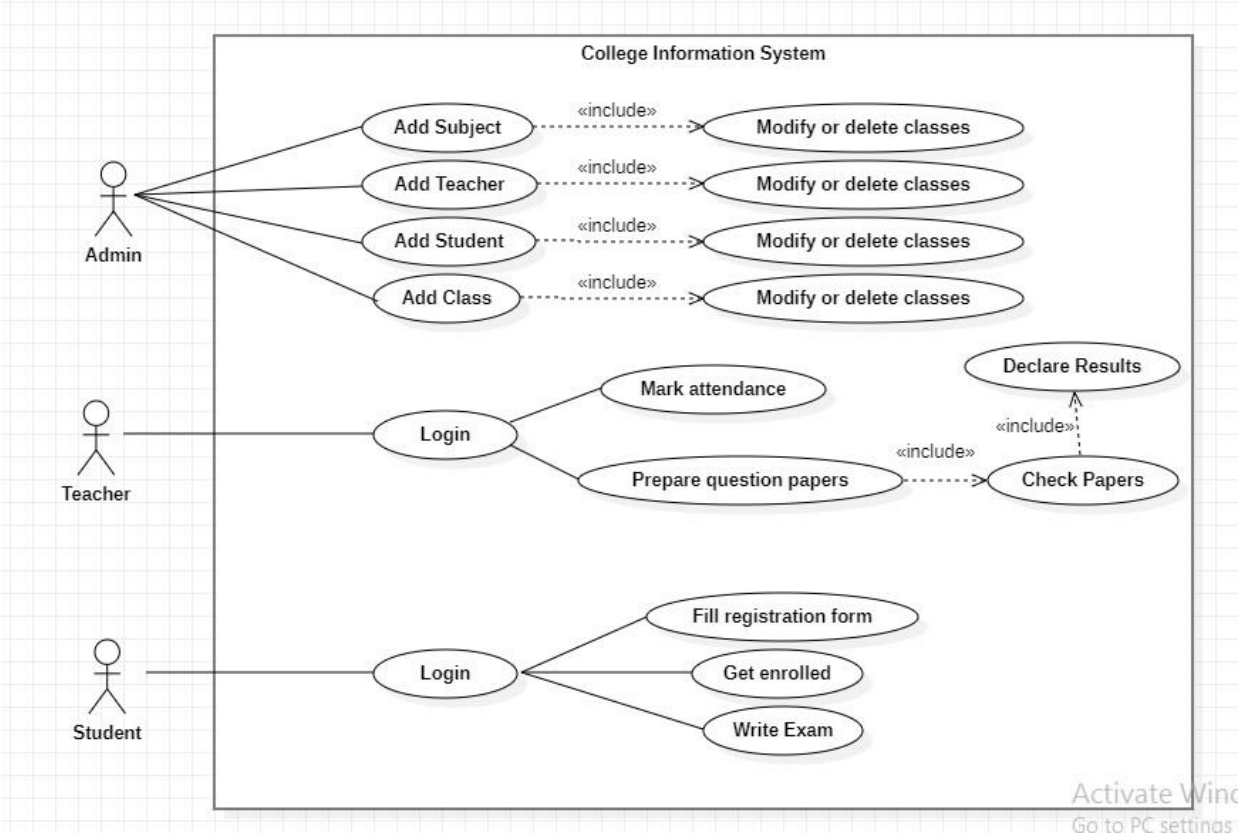

CollegeInformationSystem
click login
Logging In
userId and password entered [valid]
return
Home Page
do/show menu
select view profile
profile: ProfileView
return
click logout
Logging out
select view courses
select placements
Placement
do/display placement details
Course Details
exit
Course Enroll
do/enroll user to the selected course
enrolling
exit
Description
do/display detailed desciption of selected course
List
do/display list of courses
select a course

stm ProfileView
select view profile
Display Profile
do/show user profile with edit option
select edit
Edit
do/display all options to edit
save edits
exit
Save
do/save all fields
exit

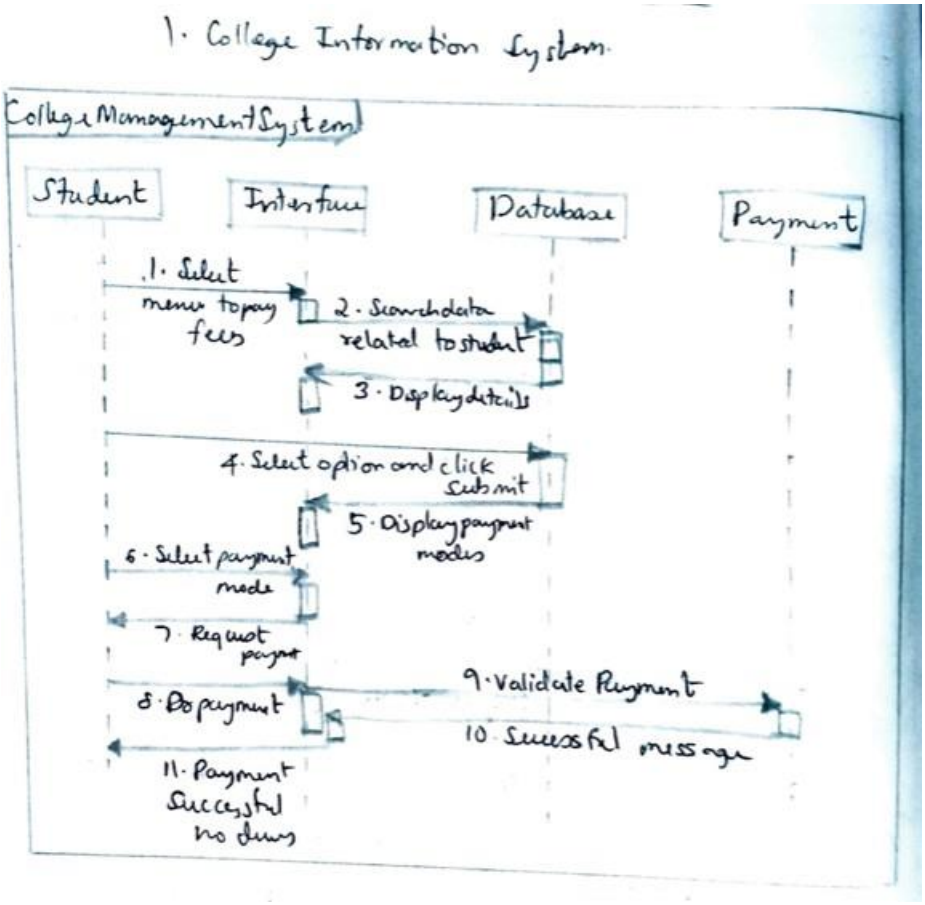
d) Advance Use Case Diagram:



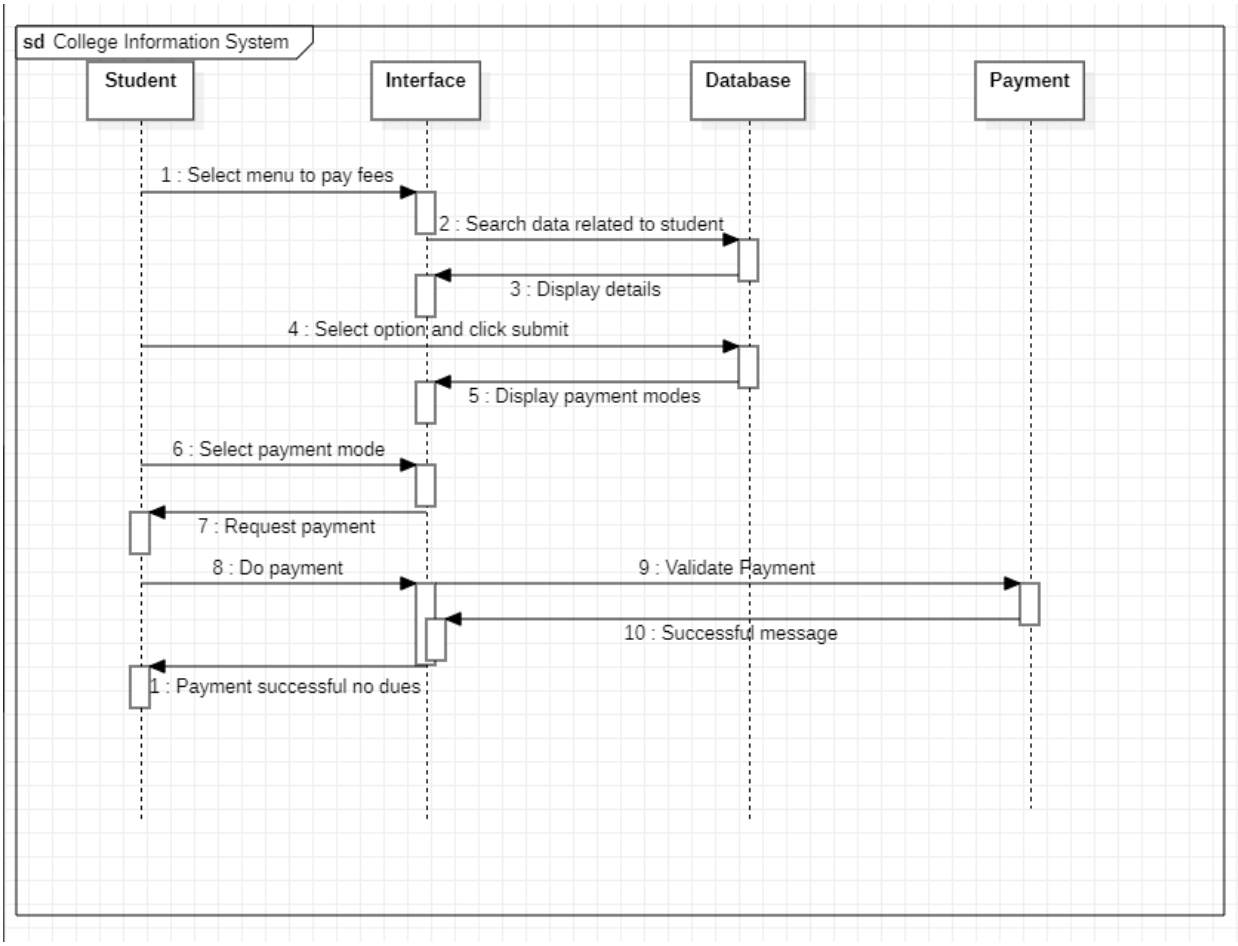
Justification: All the functionalities are mentioned in the given figure and all actors are involved in the system like student, faculty, admin, librarian and event manager. Their relations are clearly indicated in the given diagram.


College Information System
Admin
Add Subject
Add Teacher
Add Student
Add Class
«include»
Modify or delete classes
Teacher
Login
Mark attendance
Prepare question papers
Declare Results
«include»
Check Papers
Student
Login
Fill registration form
Get enrolled
Write Exam
Activate Winc
Go to PC settings

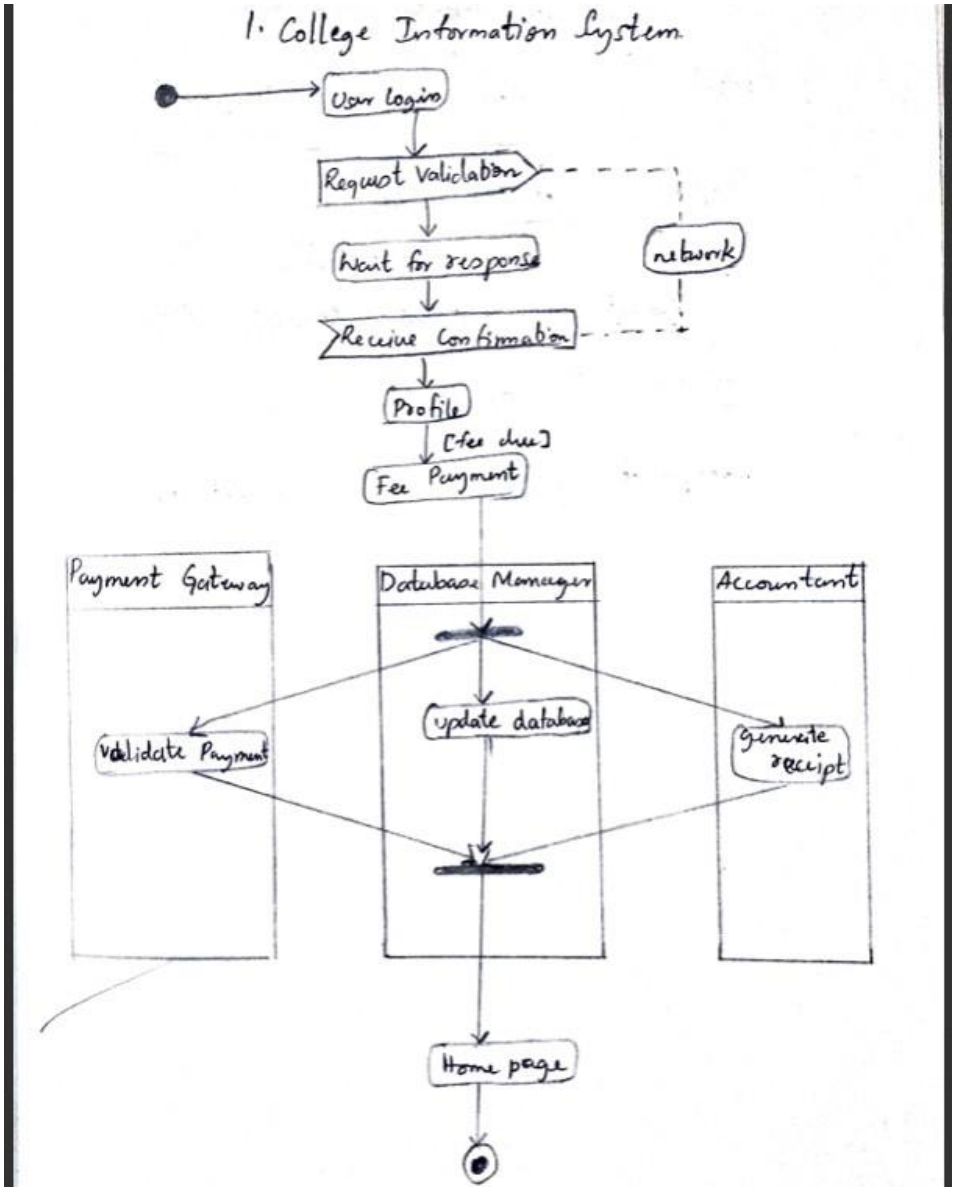
e) Sequence Diagram:



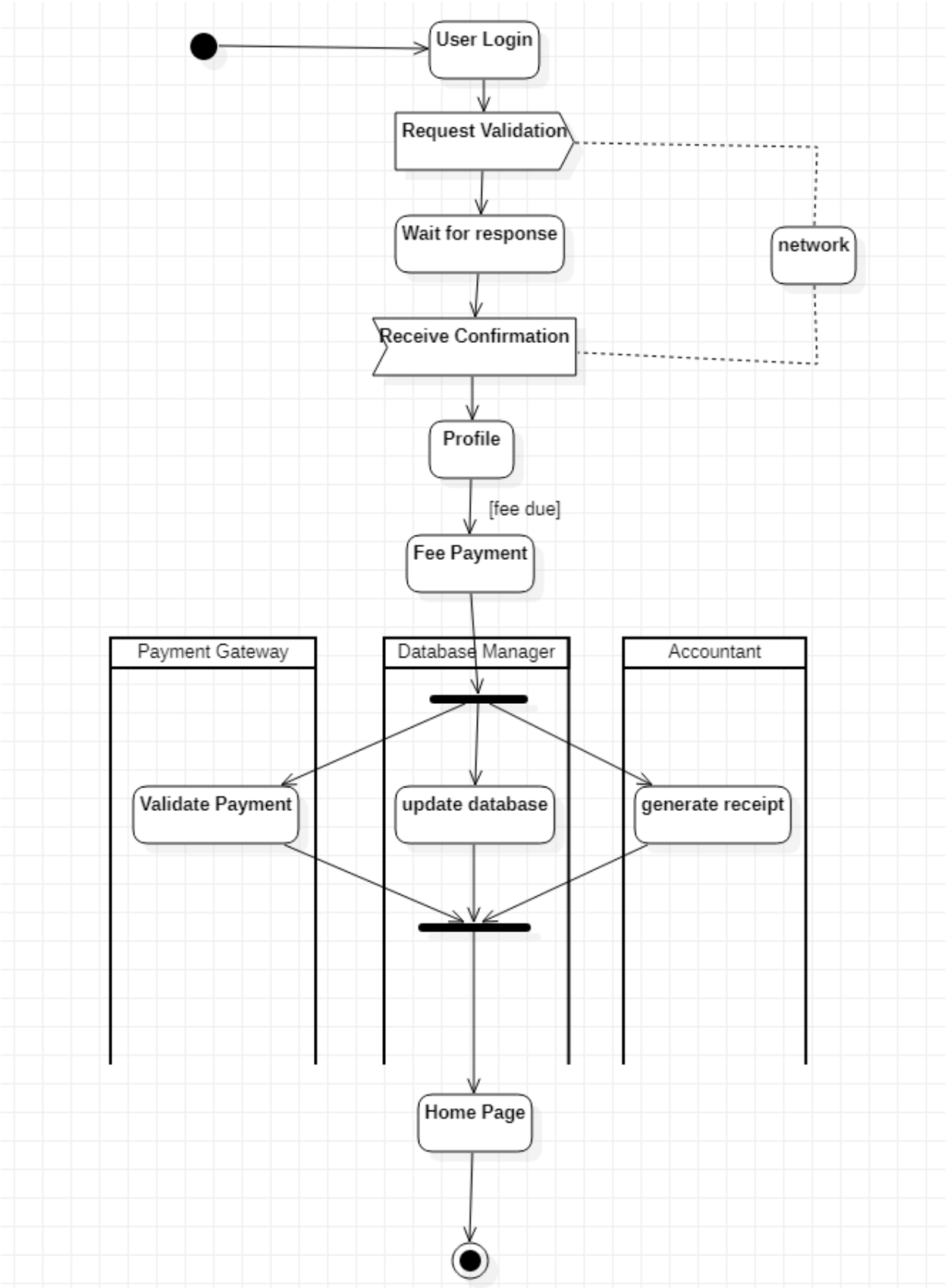
Justification: The given sequence diagram shows the complete order of all the interactions taking place between the user and user interface, database, payments and shows detailed information of payment, search and display in the system.


sd College Information System
Student
Interface
Database
Payment
1 : Select menu to pay fees
2 : Search data related to student
3 : Display details
4 : Select option and click submit
5 : Display payment modes
6 : Select payment mode
7 : Request payment
8 : Do payment
9 : Validate Payment
10 : Successful message
1 : Payment successful no dues

f) Activity Diagram:



Justification: The given diagram explains the complete working of the system from start, login to end of the session, and depicts various activities in the given application and shows the registration process of the system.


User Login
Request Validation
Wait for response
network
Receive Confirmation
Profile
[fee due]
Fee Payment
Payment Gateway
Database Manager
Accountant
Validate Payment
update database
generate receipt
Home Page

2. Hostel Management System-

a) SRS:

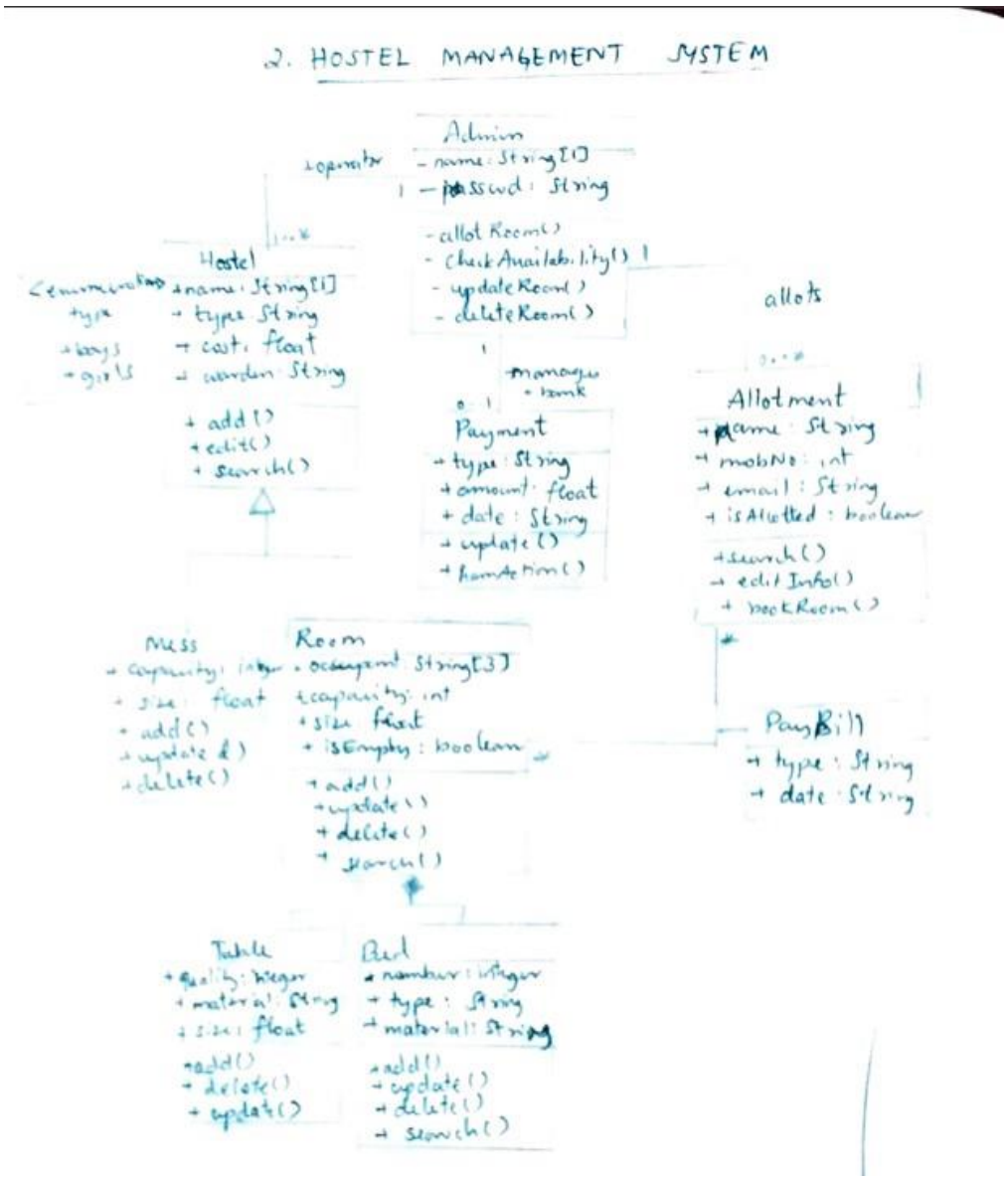
2. HOSTEL MANAGEMENT SYSTEM

Problem Statement : The hostel management system is to provide college students accomodation to the university hostel more efficiently. This project also keeps details of the hostelers and applied students. It is headed by Warden. He will be the administrator. This documents is intended to minimize human works and make hostel allocation an easier job for students and hostel authorities by providing also online application for hostel.

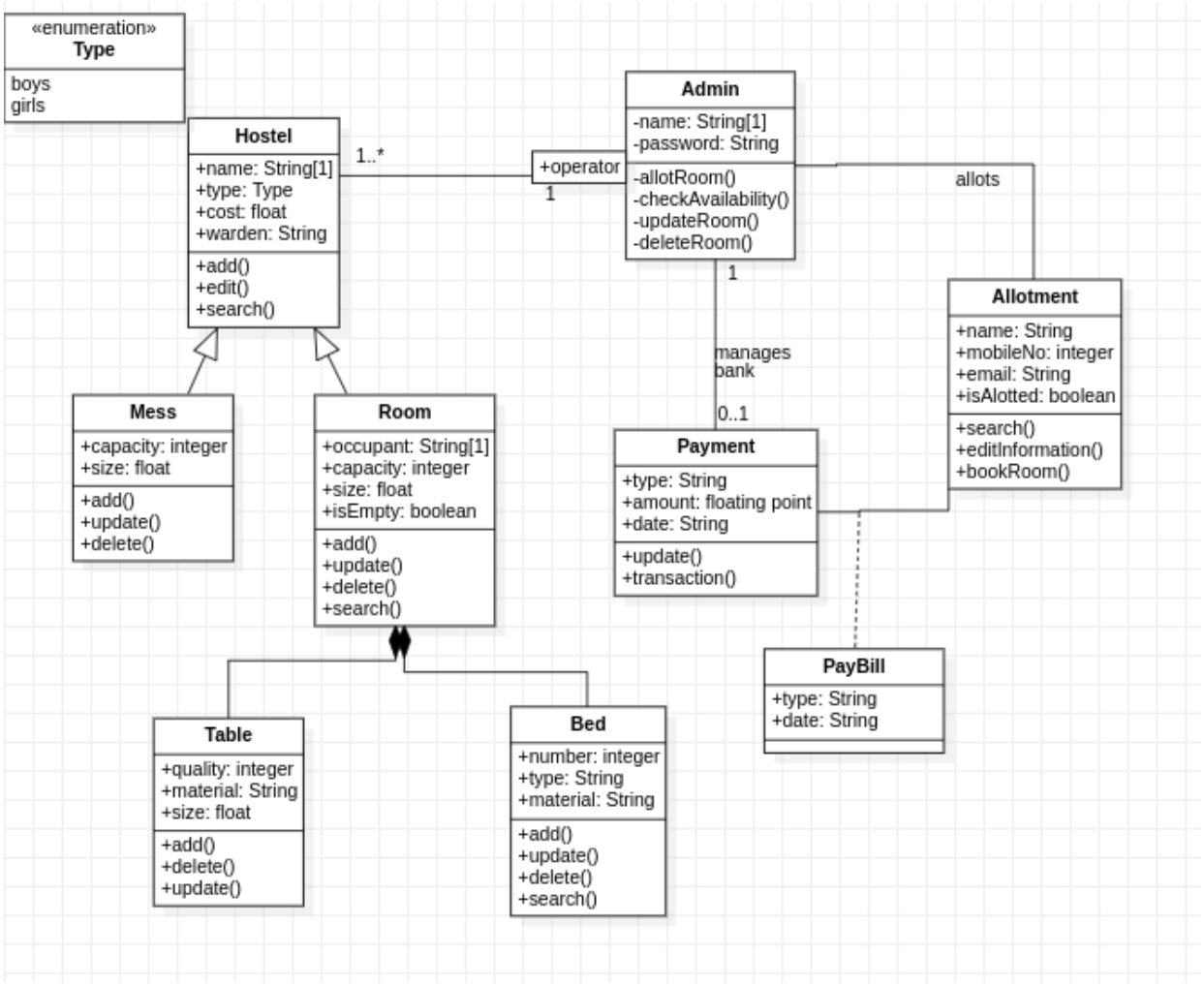
Software Requirement Specification :

- Hostel management system has admin who manages the hostel, allotes and payment methods. The admin will allocate a room to student according to the section or class. The admin will also keep track of the payment made by student/allot-es.
- As the student's course is over they will vacate their rooms. So it is required for the administrator to remove their records from the database tables.
- The allotes makes payment according to the bill generated which have the attributes bill number, type and date.
- The details of the students staying in the hostels like name, place, address, contact details is maintained in the database.
- A hostel is made up of mess and rooms. A mess account will also generate. This account having the mess status of the whole month. On the base of this account monthly, charges of mess of a student will be defined.
- The hostel management system will allow renewing the student's registration every year. The rooms of hostel are composed of table and beds, where a count of the same is maintained and the allotes can use them as they wish.

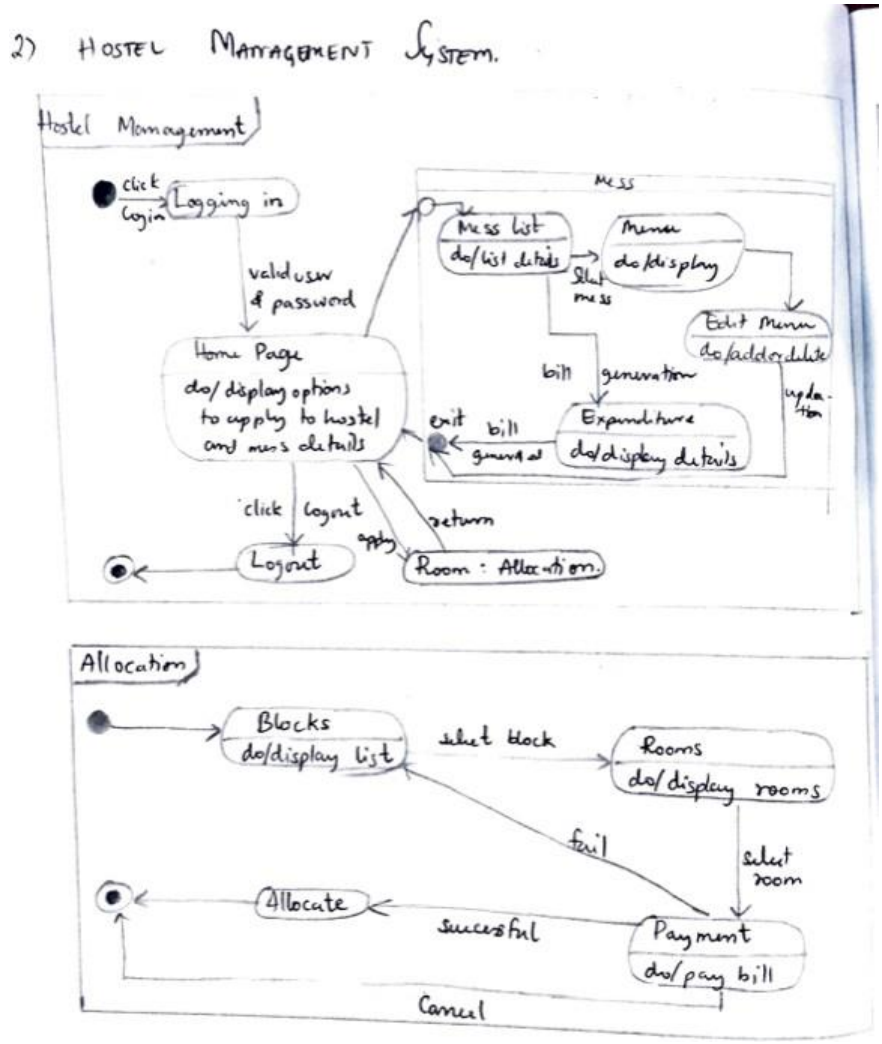
b) Advance Class Diagram:



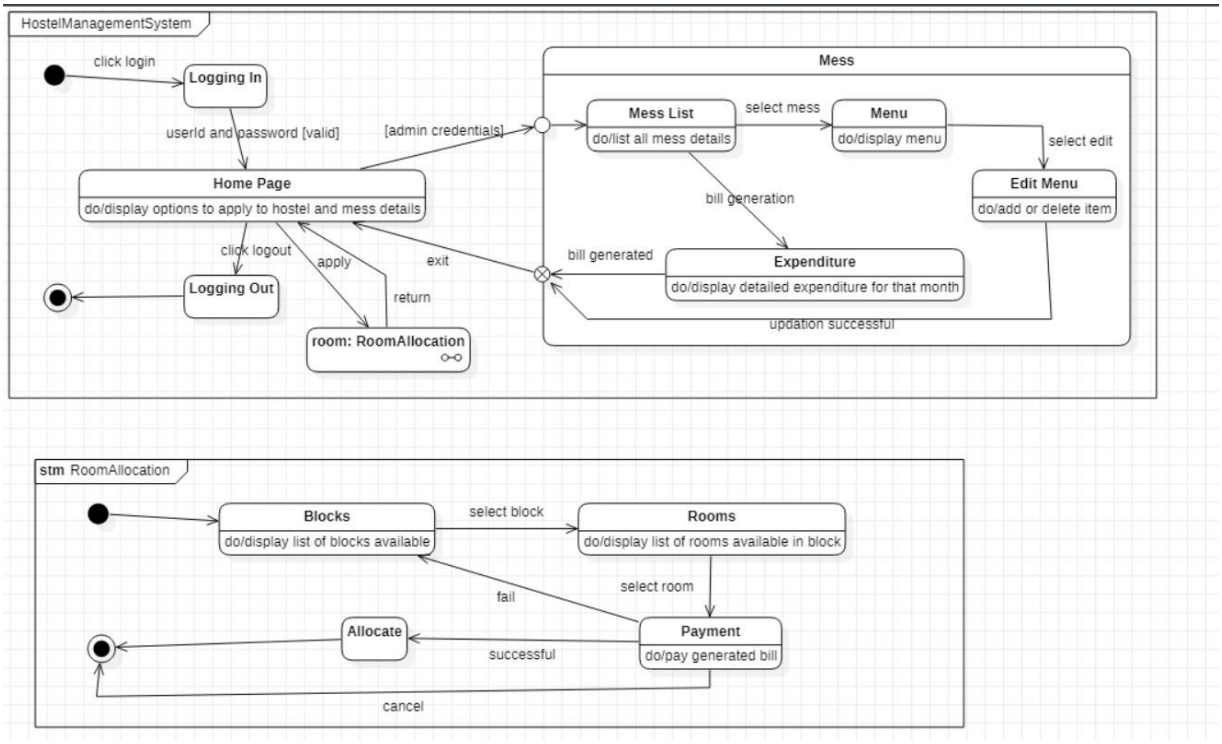
Justification: Admin controls and allotts the room, hence association, which can give feedback to the room, hence association. The students make payments through various payment methods, hence it is an enumeration. For every hostel block, a warden is assigned making it a composition.


«enumeration»
Type
boys
girls
Hostel
+name: String[1]
+type: Type
+cost: float
+warden: String
+add()
+edit()
+search()
1..*
+operator
1
Admin
-name: String[1]
-password: String
-allotRoom()
-checkAvailability()
-updateRoom()
-deleteRoom()
allots
1
manages
bank
0..1
Allotment
+name: String
+mobileNo: integer
+email: String
+isAlotted: boolean
+search()
+editInformation()
+bookRoom()
Mess
+capacity: integer
+size: float
+add()
+update()
+delete()
Room
+occupant: String[1]
+capacity: integer
+size: float
+isEmpty: boolean
+add()
+update()
+delete()
+search()
Payment
+type: String
+amount: floating point
+date: String
+update()
+transaction()
PayBill
+type: String
+date: String
Table
+quality: integer
+material: String
+size: float
+add()
+delete()
+update()
Bed
+number: integer
+type: String
+material: String
+add()
+update()
+delete()
+search()

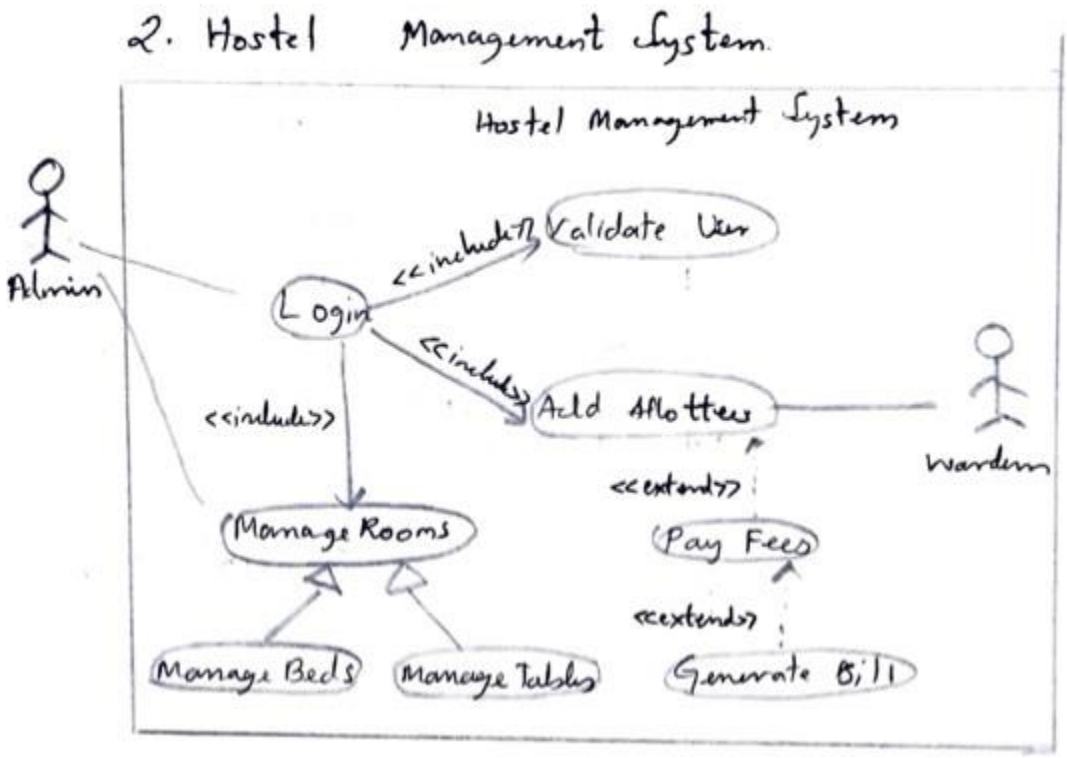
c) Advance State Diagram:



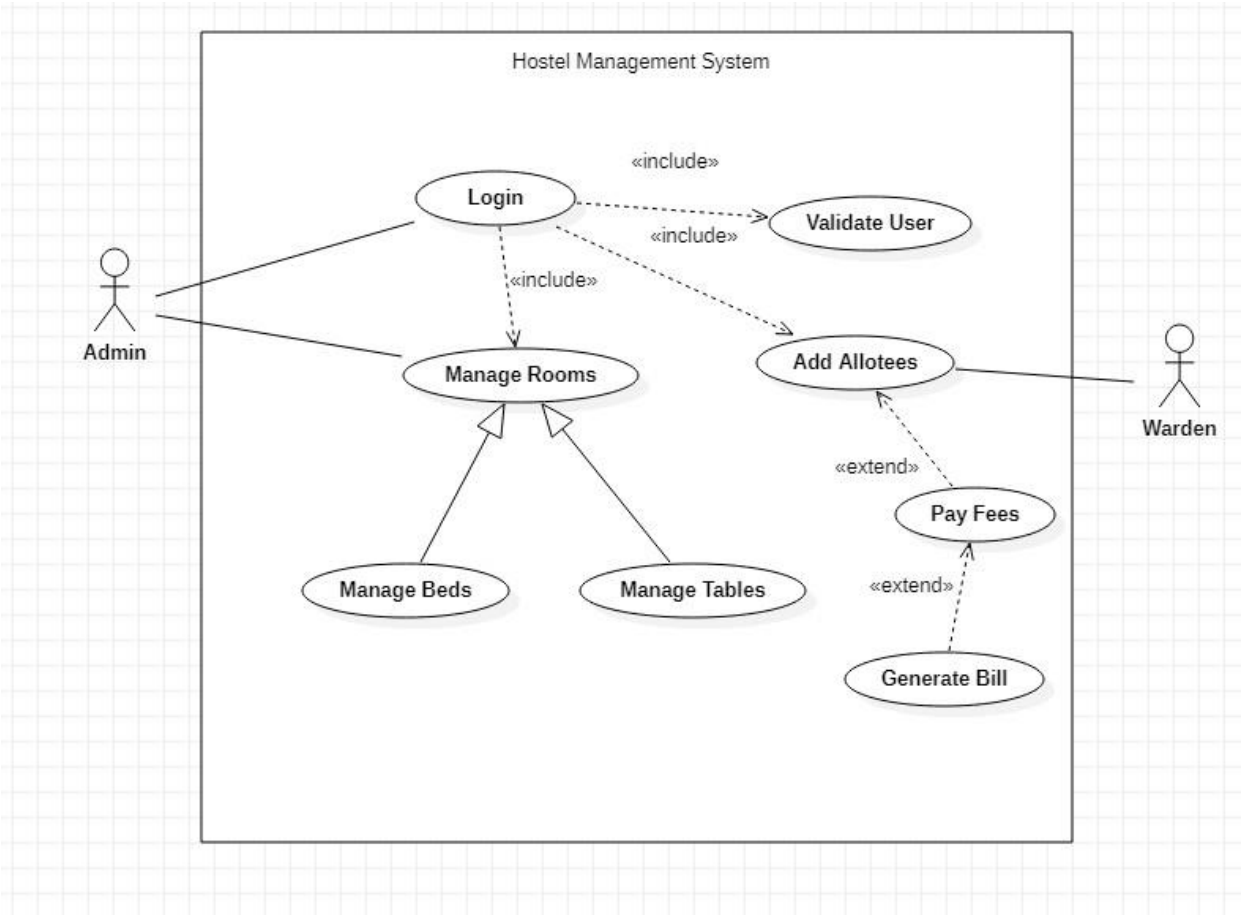
Justification: The given diagram gives a detailed description of the case with respect to a room allocation and mess menu details where it is explained with a sub machine in depth for room allocation. All the transitions and actions are mentioned.


HostelManagementSystem
click login
Logging In
userId and password [valid]
[admin credentials]
Home Page
do/display options to apply to hostel and mess details
click logout
apply
exit
return
Logging Out
room: RoomAllocation
Mess
Mess List
do/list all mess details
select mess
Menu
do/display menu
select edit
Edit Menu
do/add or delete item
bill generation
bill generated
Expenditure
do/display detailed expenditure for that month
updation successful
stm RoomAllocation
Blocks
do/display list of blocks available
select block
Rooms
do/display list of rooms available in block
select room
fail
Payment
do/pay generated bill
Allocate
successful
cancel

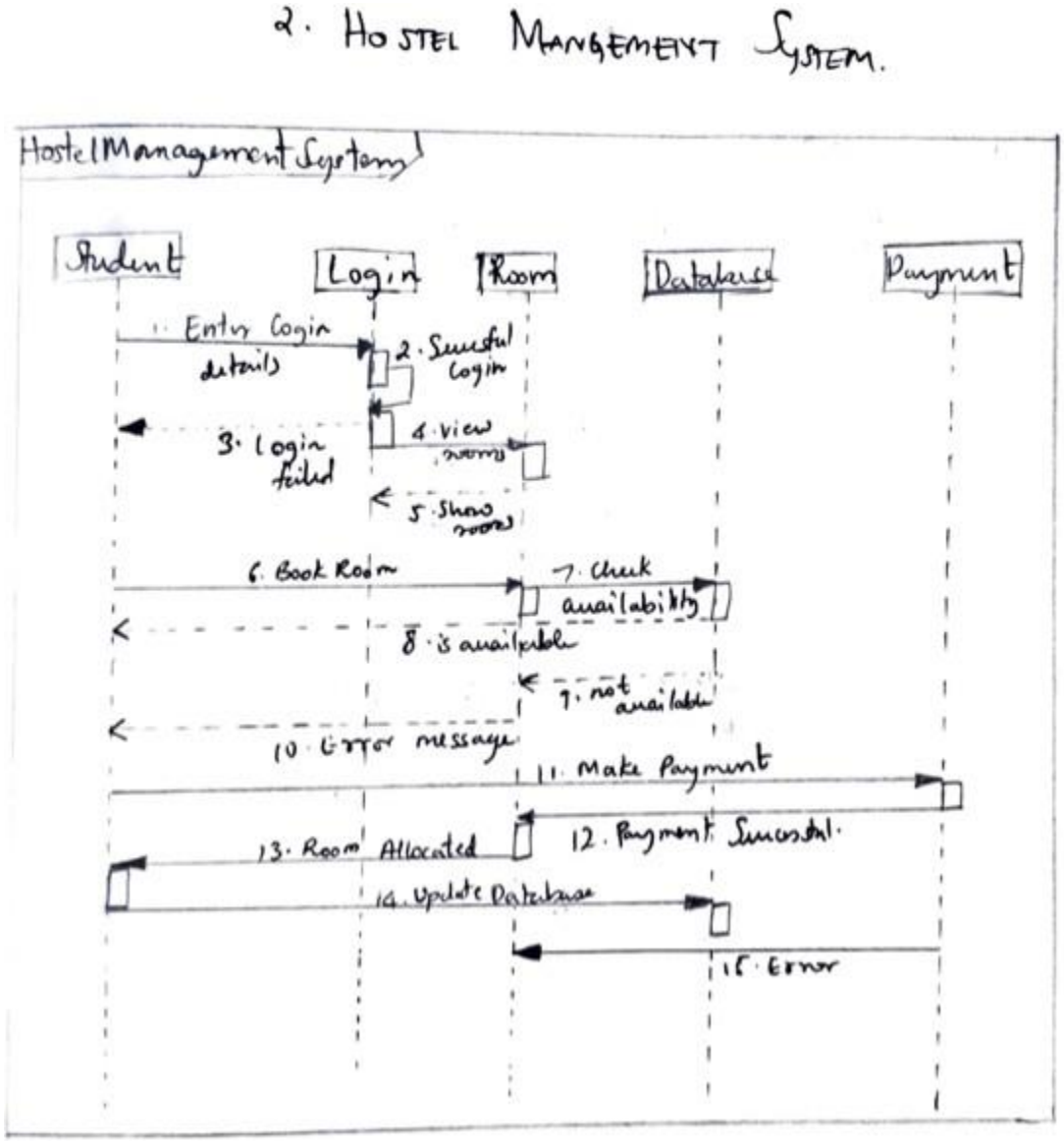
d) Advance Use Case Diagram:



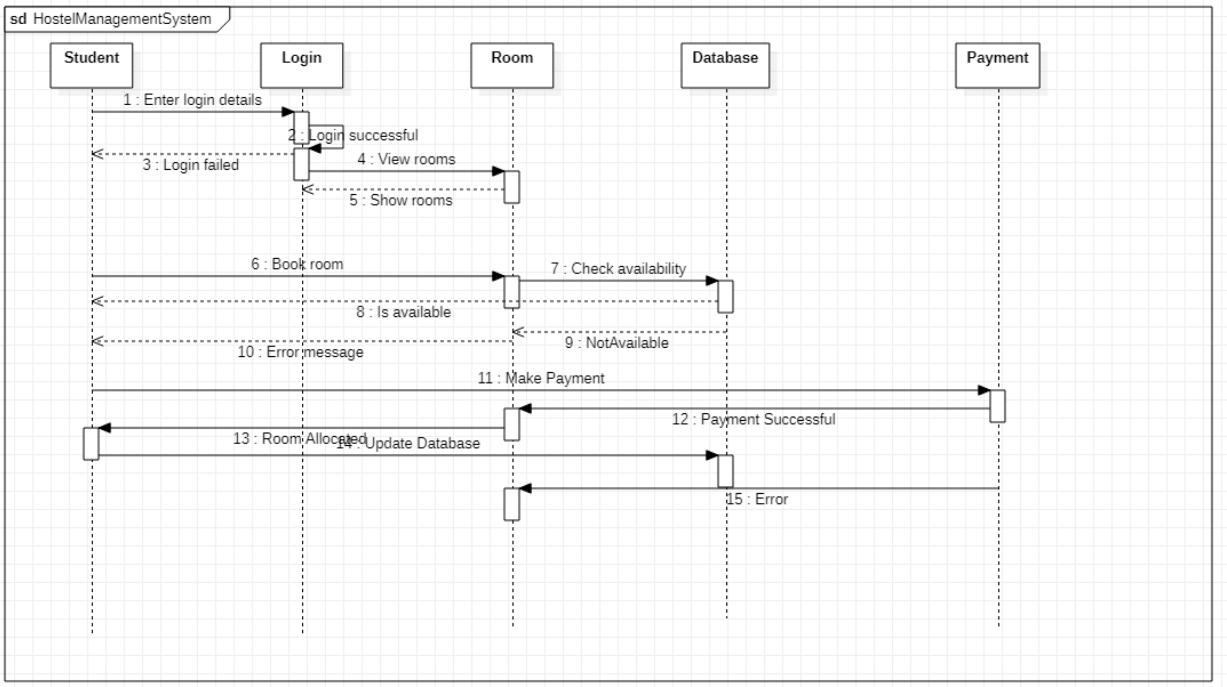
Justification: For the given case, all the functionalities are accounted for and respective actors are given the right functionalities. All the roles and tasks have been allotted.


Hostel Management System
«include»
Login
Validate User
«include»
«include»
Admin
Manage Rooms
Add Allotees
Warden
«extend»
Pay Fees
Manage Beds
Manage Tables
«extend»
Generate Bill

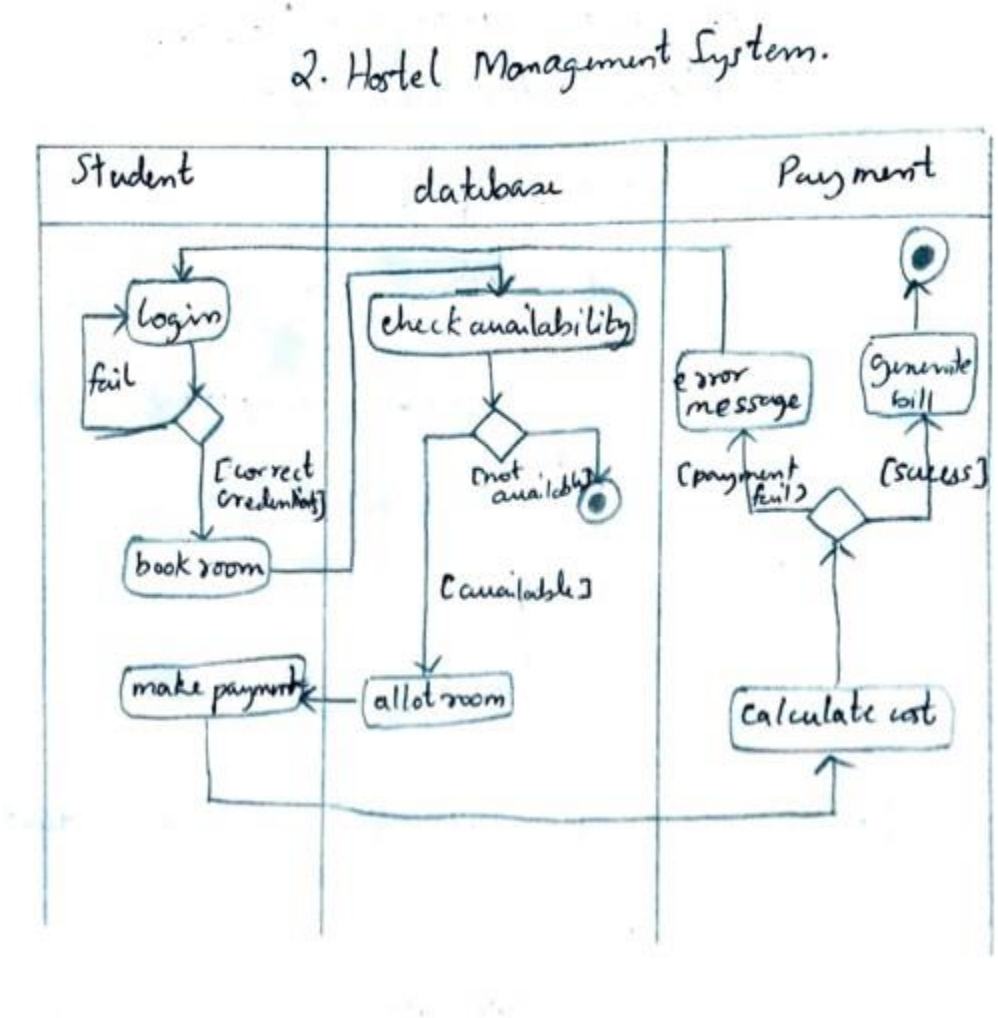
e) Sequence Diagram:



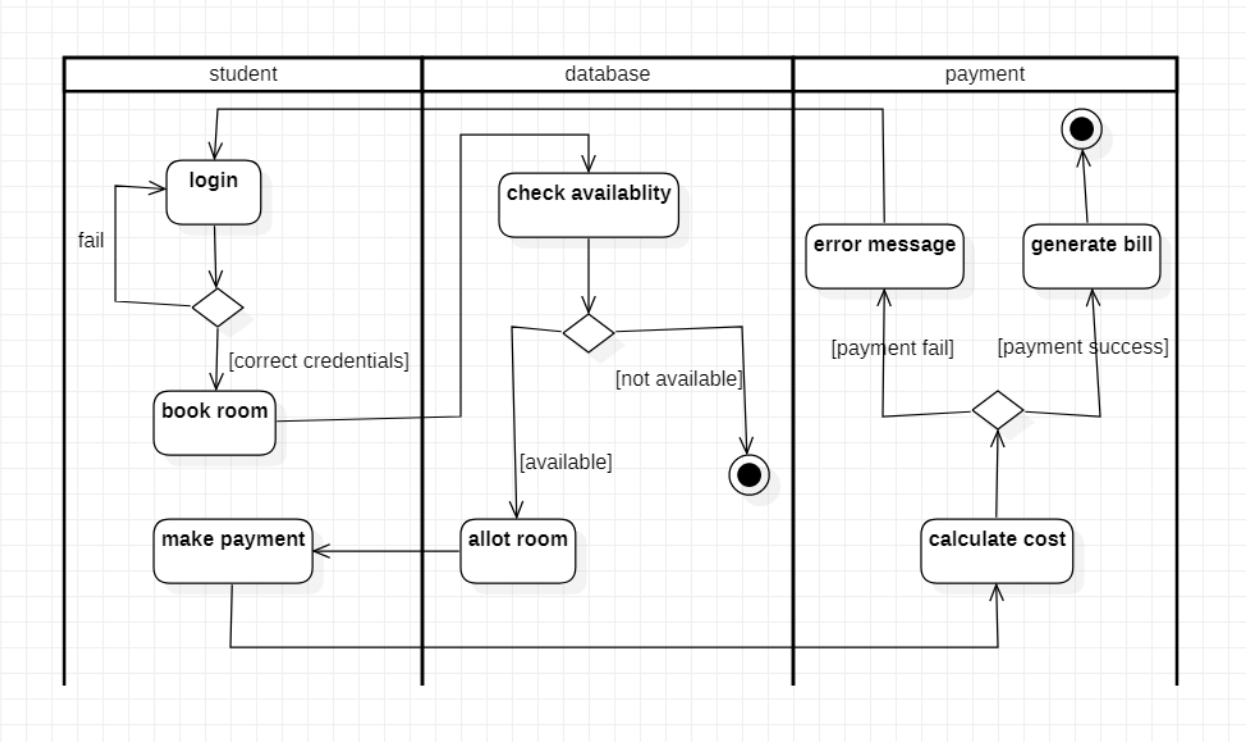
Justification: The given model shows the complete steps in the whole process of registration, allocation of rooms. The payment procedure is also shown and accounted for.


sd HostelManagementSystem
Student
Login
Room
Database
Payment
1 : Enter login details
2 : Login successful
3 : Login failed
4 : View rooms
5 : Show rooms
6 : Book room
7 : Check availability
8 : Is available
9 : NotAvailable
10 : Error message
11 : Make Payment
12 : Payment Successful
13 : Room Allocated
14 : Update Database
15 : Error

f) Activity Diagram:

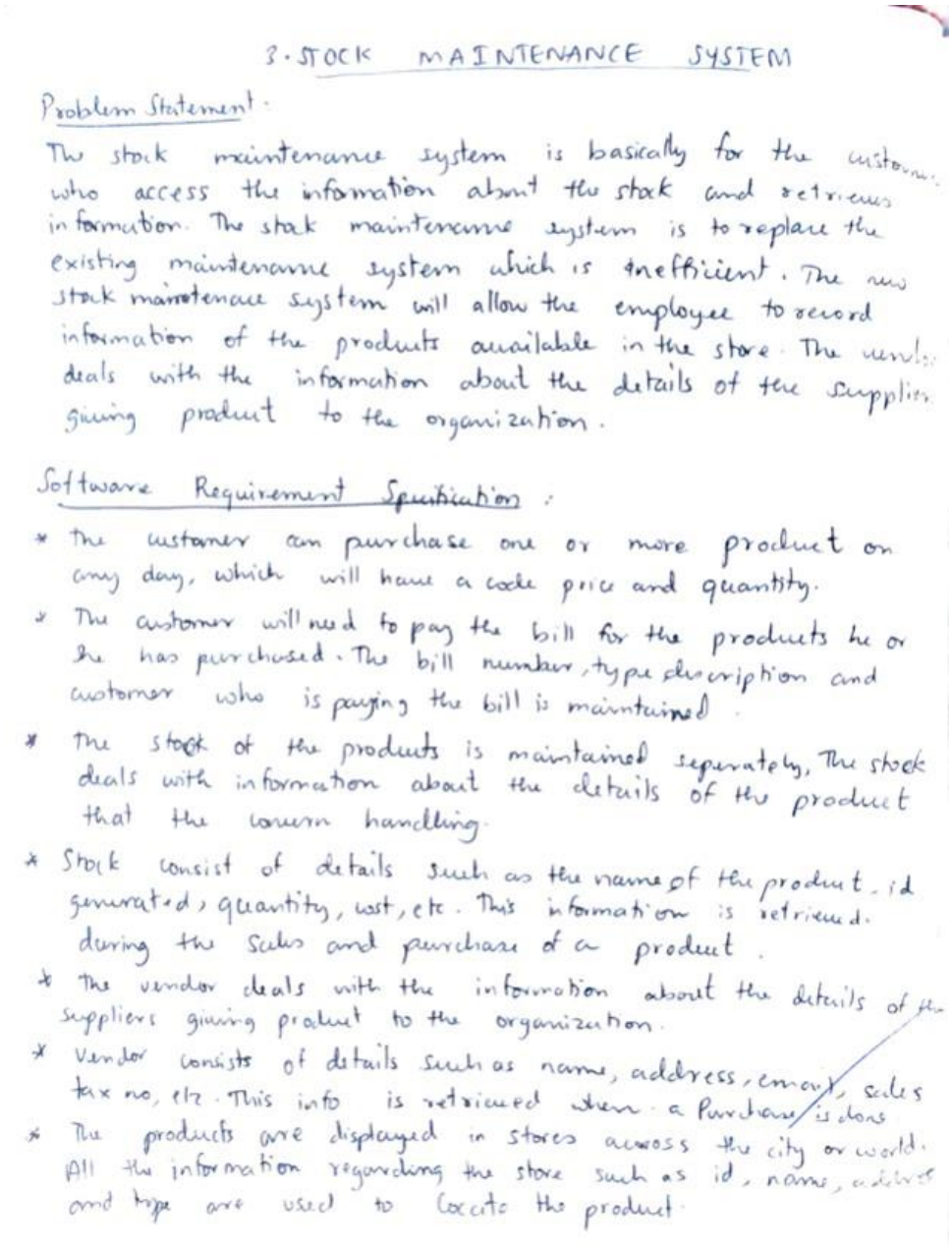


Justification: The given figure explains the working of the application system with all the functions and interactions with respect to the actors involved in the system. It accounts for modifications and additions on the database and payment procedures.


student
database
payment
login
fail
[correct credentials]
book room
check availablity
[not available]
[available]
allot room
make payment
error message
generate bill
[payment fail]
[payment success]
calculate cost

3. Stock Maintenance System-

a) SRS:



3. STOCK MAINTENANCE SYSTEM

Problem Statement:

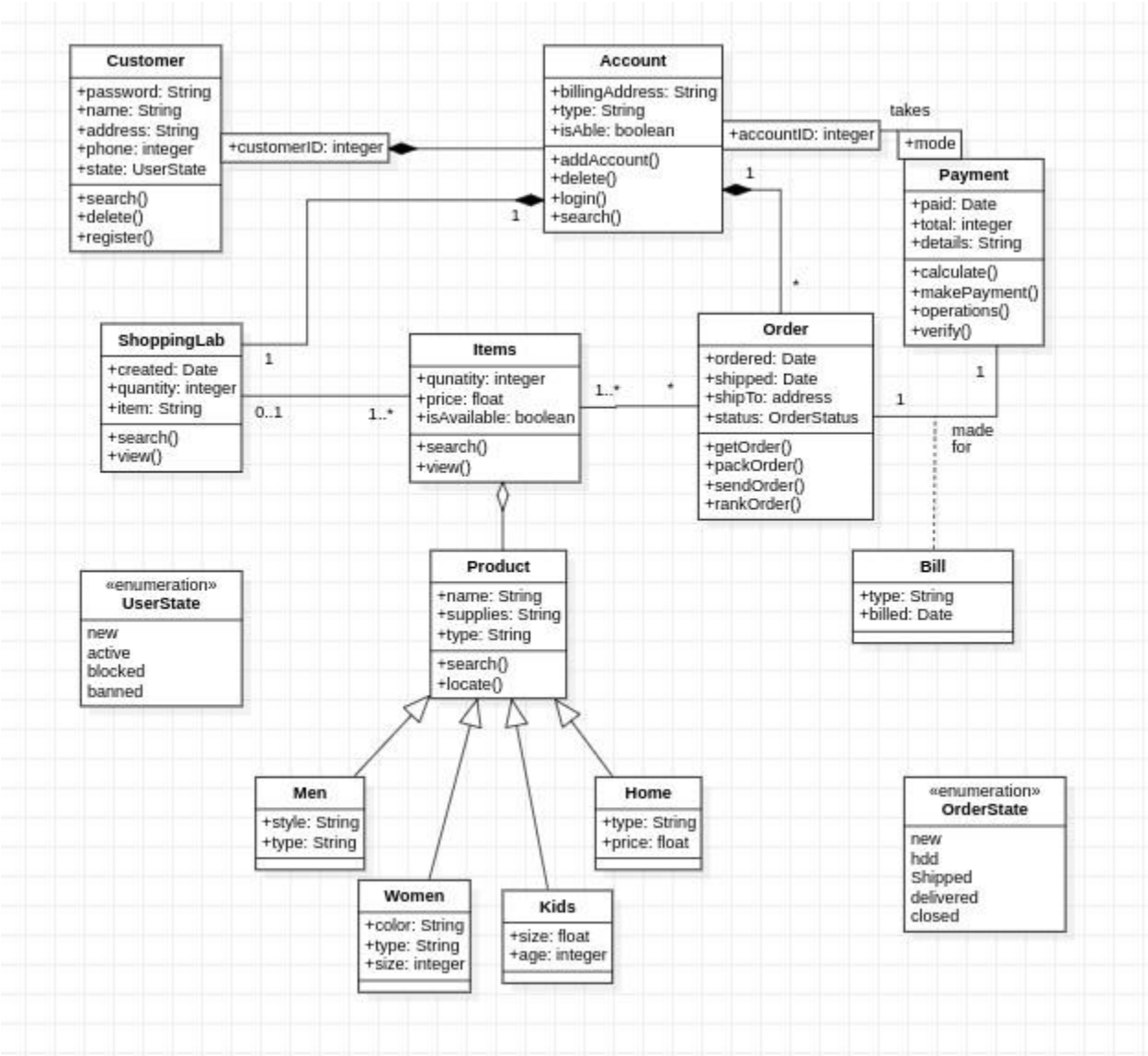
The stock maintenance system is basically for the customer who access the information about the stock and retrieves information. The stock maintenance system is to replace the existing maintenance system which is inefficient. The new stock maintenance system will allow the employee to record information of the products available in the store. The vendor deals with the information about the details of the suppliers giving product to the organization.

Software Requirement Specification:

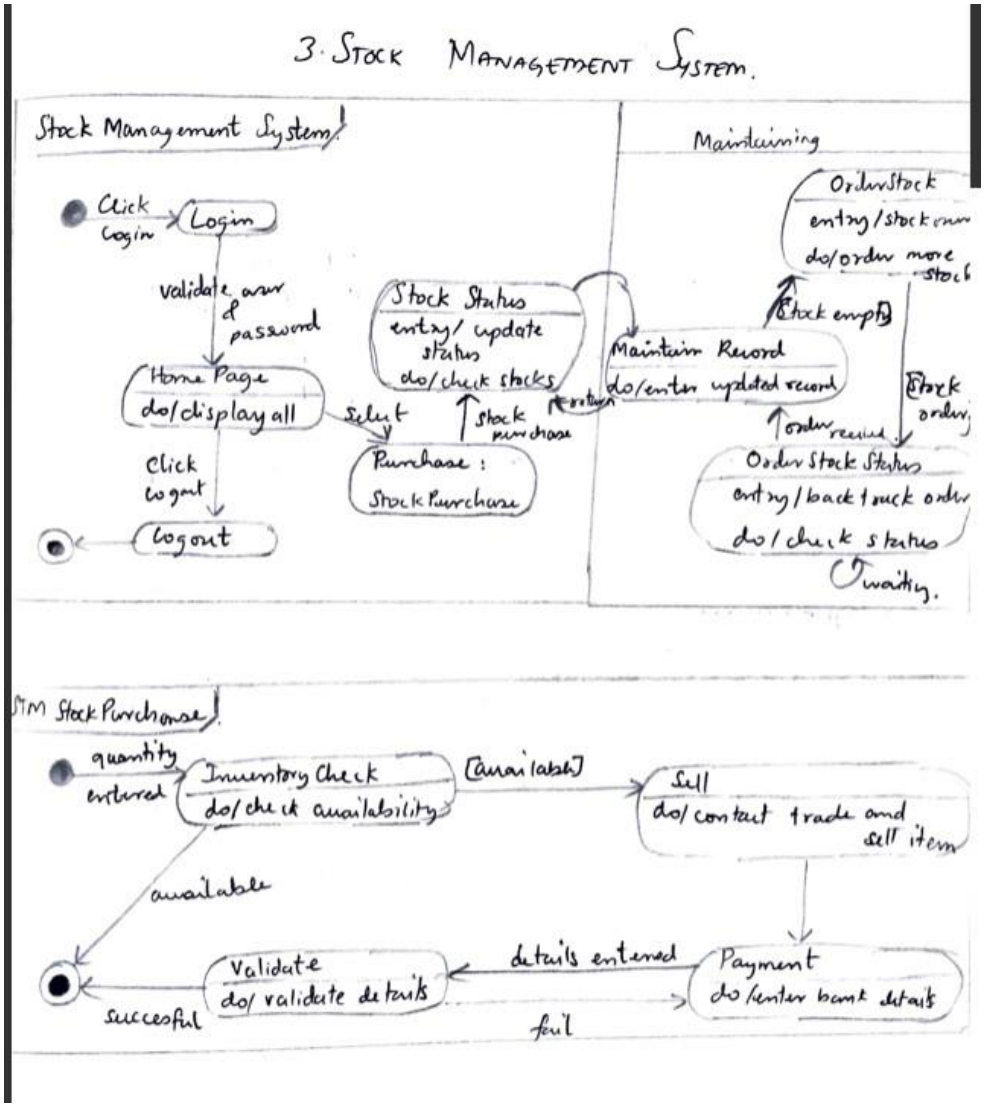
* The customer can purchase one or more product on any day, which will have a code price and quantity.
* The customer will need to pay the bill for the products he or she has purchased. The bill number, type description and customer who is paying the bill is maintained.
* The stock of the products is maintained seperately, The stock deals with information about the details of the product that the concern handling.
* Stock consist of details such as the name of the product, id generated, quantity, cost, etc. This information is retrieved during the sales and purchase of a product.
* The vendor deals with the information about the details of the suppliers giving product to the organization.
* Vendor consists of details such as name, address, email, sales tax no, etc. This info is retrieved when a Purchase is done
* The products are displayed in stores across the city or world. All the information regarding the store such as id, name, address and type are used to locate the product.

b) Advance Class Diagram:

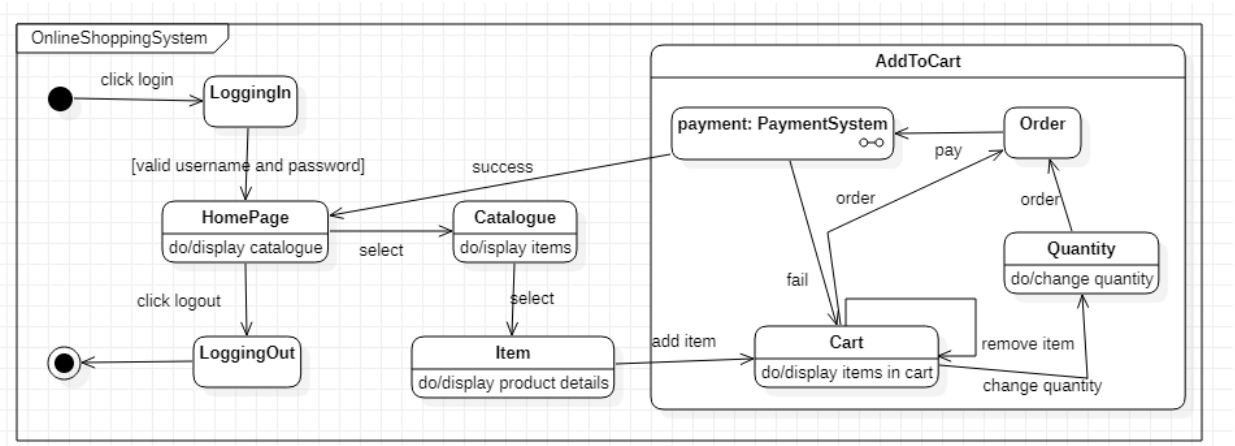
Justification: A store has several substores generalized and composed of many products which are in turn associated with vendors. Every customer buys a product hence associated and pays the bill for stock purchased. All the stocks are aggregated to the product.

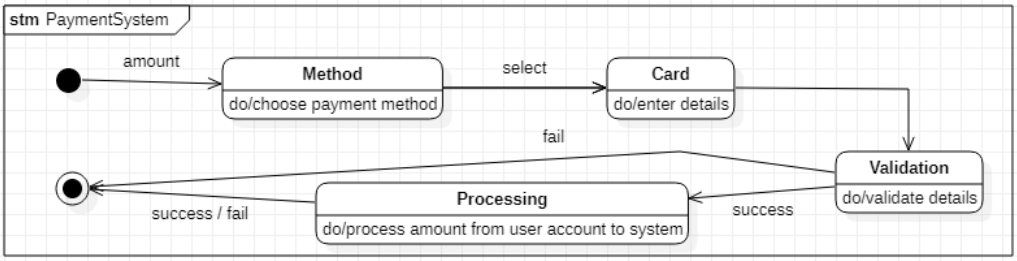

Customer
+password: String
+name: String
+address: String
+phone: integer
+state: UserState
+search()
+delete()
+register()
+customerID: integer
Account
+billingAddress: String
+type: String
+isAble: boolean
+addAccount()
+delete()
+login()
+search()
+accountID: integer
takes
+mode
Payment
+paid: Date
+total: integer
+details: String
+calculate()
+makePayment()
+operations()
+verify()
1
1
*
1
made
for
ShoppingLab
+created: Date
+quantity: integer
+item: String
+search()
+view()
1
0..1
1..*
Items
+qunatity: integer
+price: float
+isAvailable: boolean
+search()
+view()
1..*
*
Order
+ordered: Date
+shipped: Date
+shipTo: address
+status: OrderStatus
+getOrder()
+packOrder()
+sendOrder()
+rankOrder()
«enumeration»
UserState
new
active
blocked
banned
Product
+name: String
+supplies: String
+type: String
+search()
+locate()
Bill
+type: String
+billed: Date
Men
+style: String
+type: String
Women
+color: String
+type: String
+size: integer
Kids
+size: float
+age: integer
Home
+type: String
+price: float
«enumeration»
OrderState
new
hdd
Shipped
delivered
closed

c) Advance State Diagram:

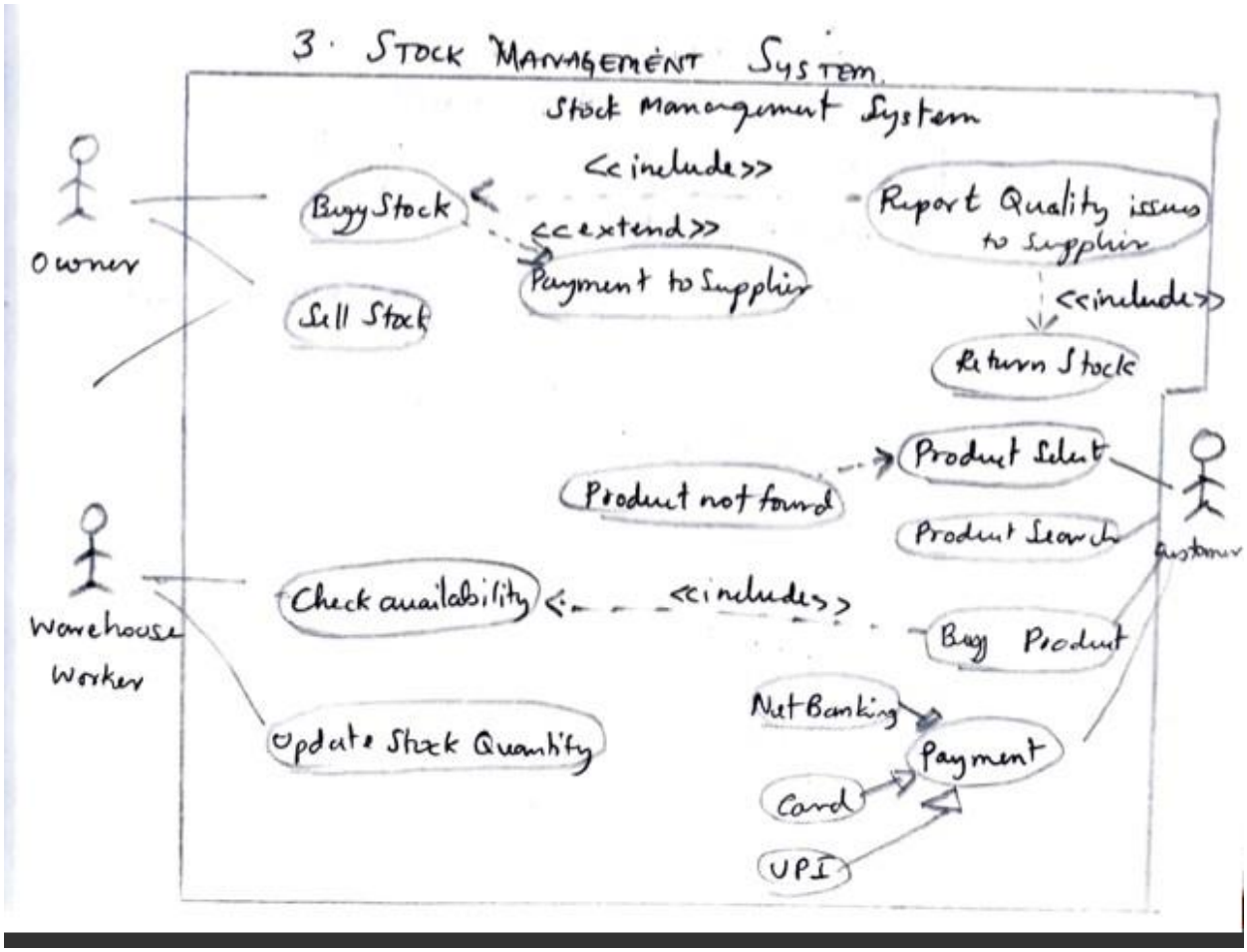


Justification: This machine explains a scenario of maintaining stocks and order of purchase. The details of the management are mentioned in the sub machine and all the respective transition and conditions are mentioned with all activities in detail.

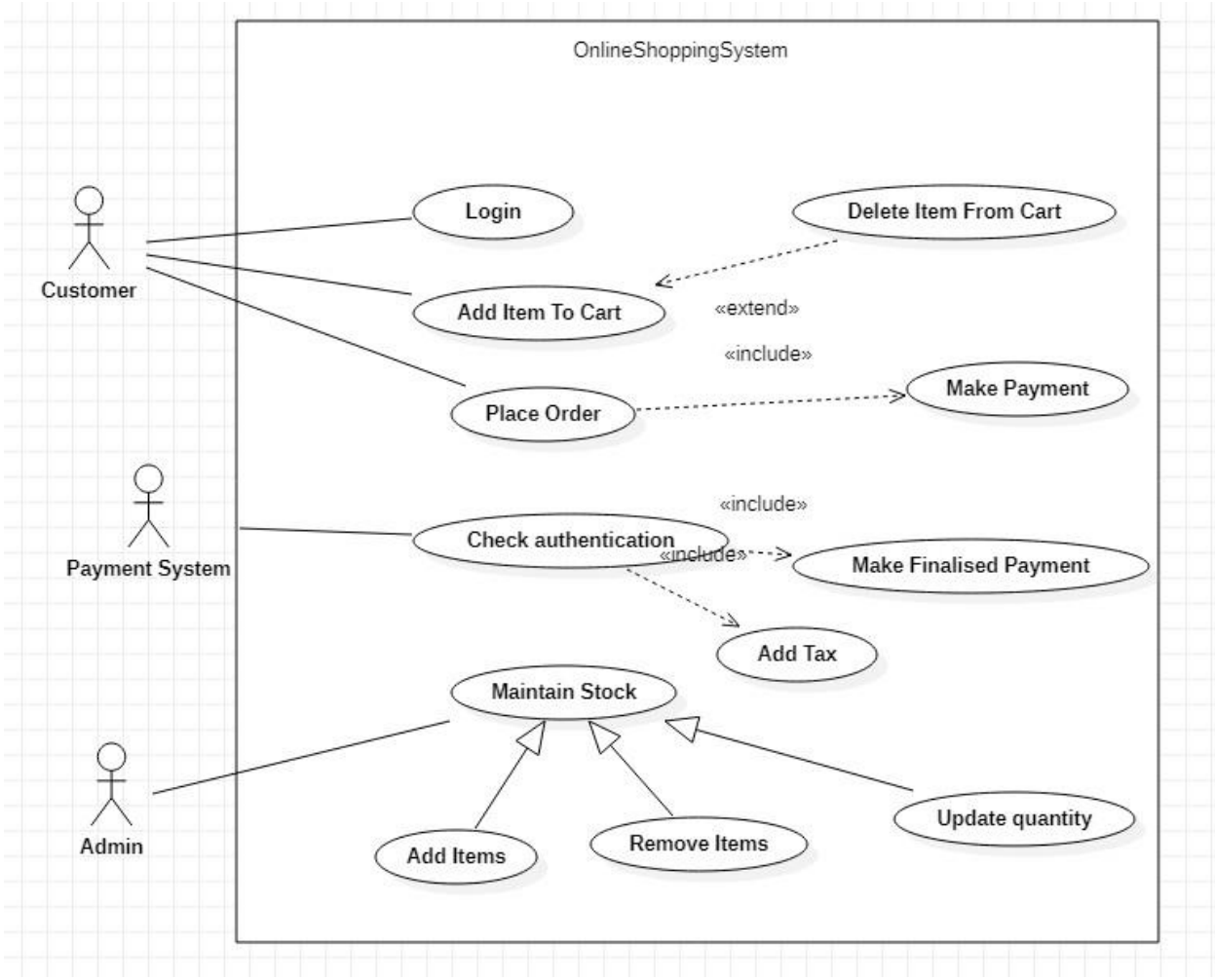

OnlineShoppingSystem
click login
LoggingIn
[valid username and password]
HomePage
do/display catalogue
select
Catalogue
do/isplay items
click logout
LoggingOut
select
Item
do/display product details
AddToCart
payment: PaymentSystem
success
Order
pay
order
order
Quantity
do/change quantity
fail
add item
Cart
do/display items in cart
remove item
change quantity


stm PaymentSystem
amount
Method
do/choose payment method
select
Card
do/enter details
fail
Validation
do/validate details
success / fail
Processing
do/process amount from user account to system
success

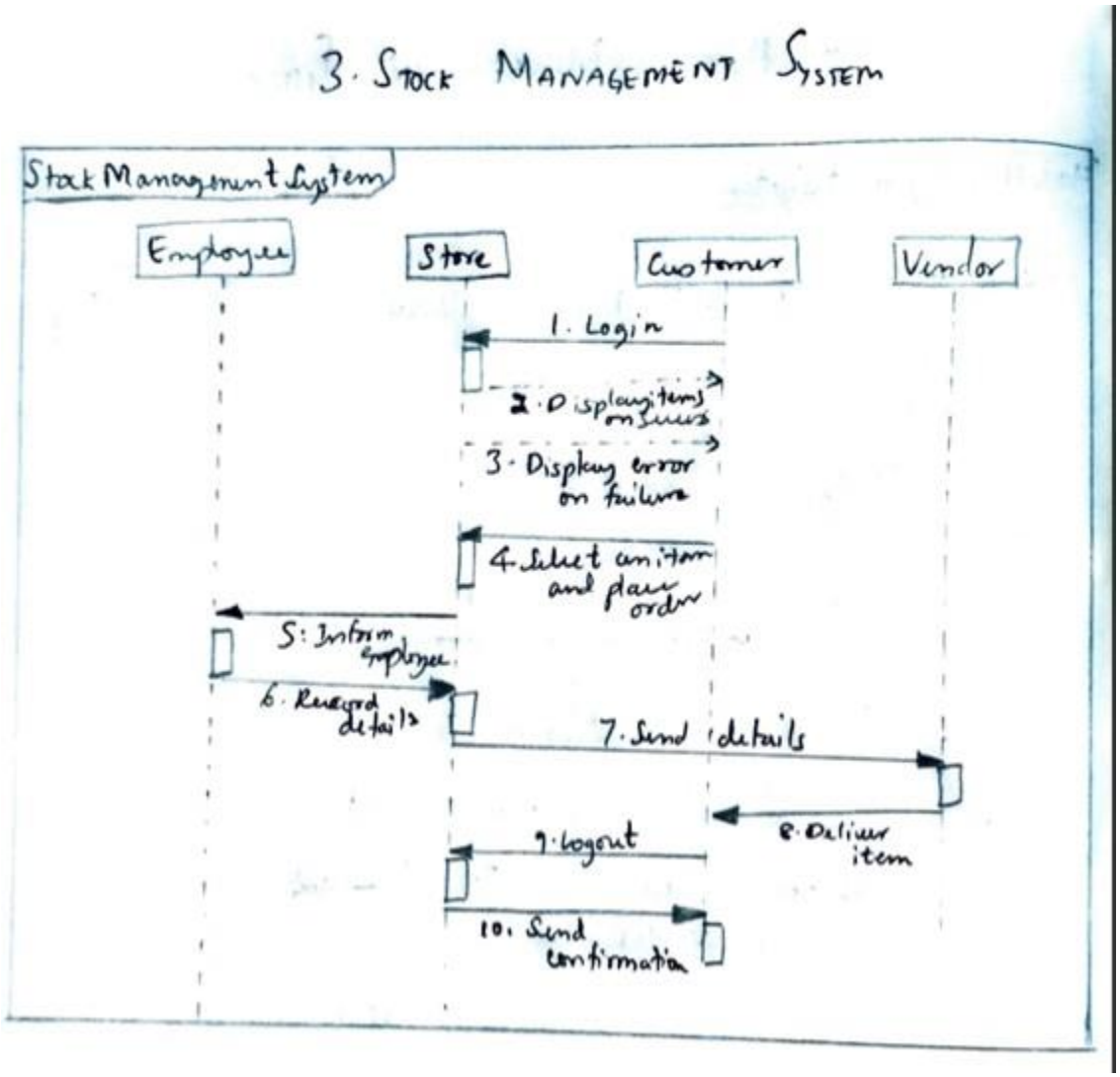
d) Advance Use Case Diagram:



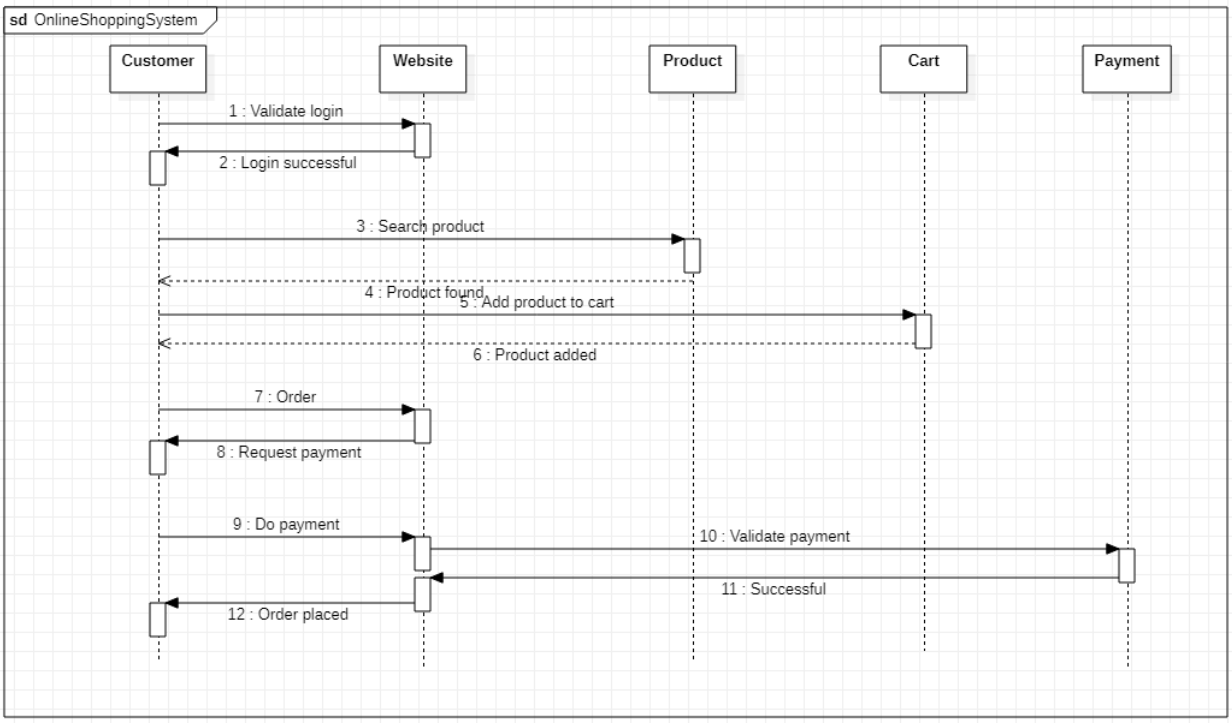
Justification: All function for the given system are mentioned and actors such as customer, worker and owner are assigned appropriate roles and functionalities.


OnlineShoppingSystem
Customer
Login
Delete Item From Cart
Add Item To Cart
«extend»
«include»
Place Order
Make Payment
Payment System
«include»
Check authentication
«include»
Make Finalised Payment
Add Tax
Maintain Stock
Admin
Add Items
Remove Items
Update quantity

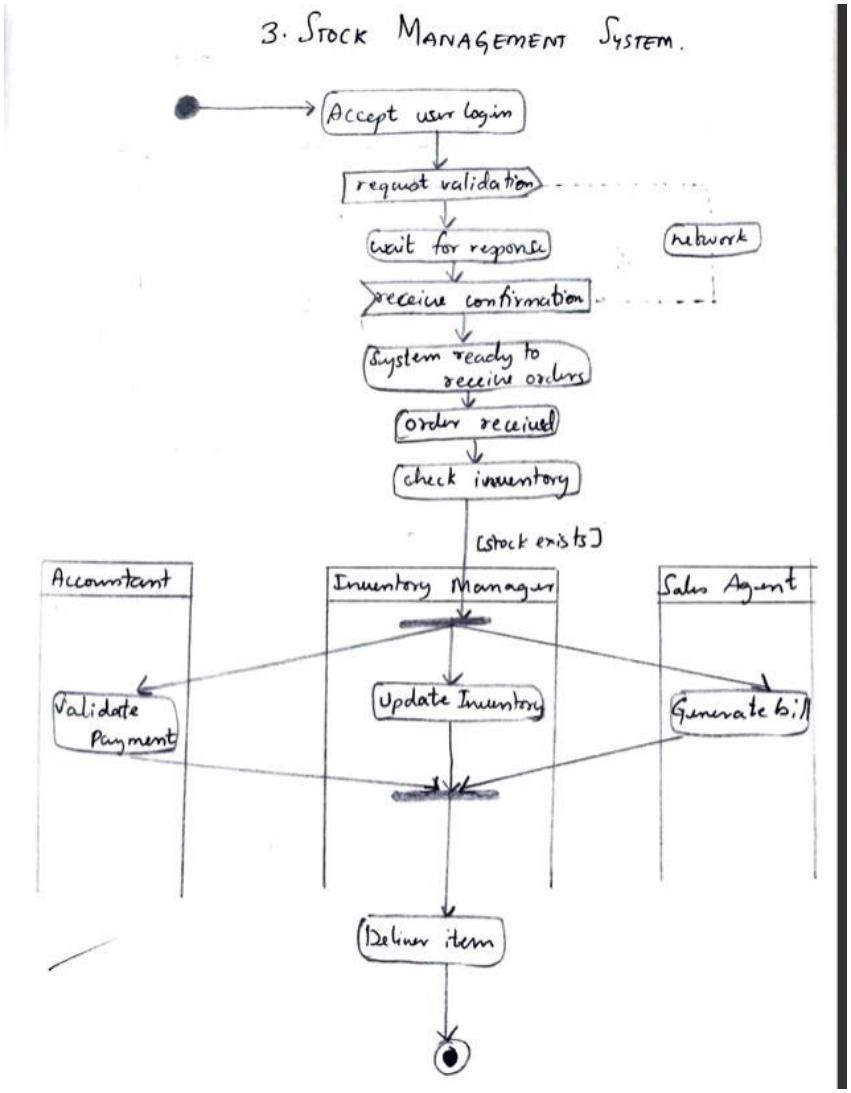
e) Sequence Diagram:



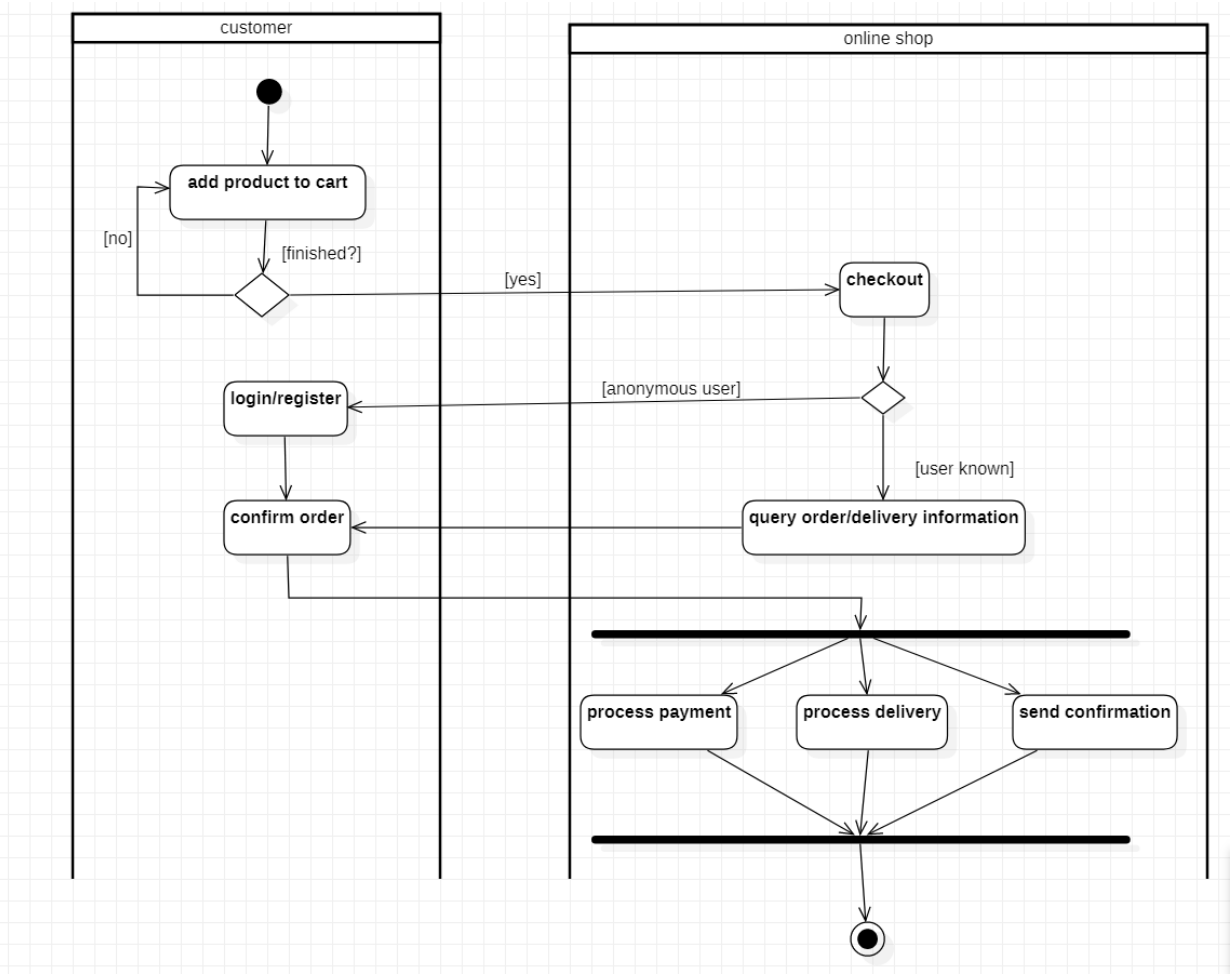
Justification: The given model shows the complete interactions of the customer with the vendor and all the steps in the process of login, search and selecting the stocks and purchasing them. All fields are updated accordingly


sd OnlineShoppingSystem
Customer
Website
Product
Cart
Payment
1 : Validate login
2 : Login successful
3 : Search product
4 : Product found
5 : Add product to cart
6 : Product added
7 : Order
8 : Request payment
9 : Do payment
10 : Validate payment
11 : Successful
12 : Order placed

f) Activity Diagram:



Justification: The given activity diagram gives complete system interaction with user, explains from user's login, validation of credentials and management activities.


customer
online shop
add product to cart
[no]
[finished?]
[yes]
checkout
[anonymous user]
login/register
[user known]
confirm order
query order/delivery information
process payment
process delivery
send confirmation

4. Coffee Vending Machine-

a) SRS:

4. COFFEE VENDING MACHINE

Problem Statement:

The coffee vending machine is basically for the customers to buy coffee by themselves without any third person being involved. A coffee vending machine sells different types of coffee such as cappuccino, latte, etc. Each type of coffee has a price and a name. A customer can buy their choice of coffee by selecting the button of their coffee and paying for the same through the coin box.

Software Requirements Specification.

*) The vending machine must have money box, coin slot, display screen and products ie coffee to the user.

*) The user on selecting a coffee, the coffee machine must be able to dispense the selected coffee to the user.

*) The user shall get empty cup placed right below the filter. The user shall be able to choose his preferred beverage from the list of options

*) There must be buttons for user to interact with the system.

*) The user shall be able to purchase one kind of available drink at a time and get back the exact changes if he has put extra money. The User shall be able to quit the dispense of any beverage at any time during dispensing.

*) The system shall check for properly inserted coins

*) The system shall be able to dispense coffee after coin insertion.

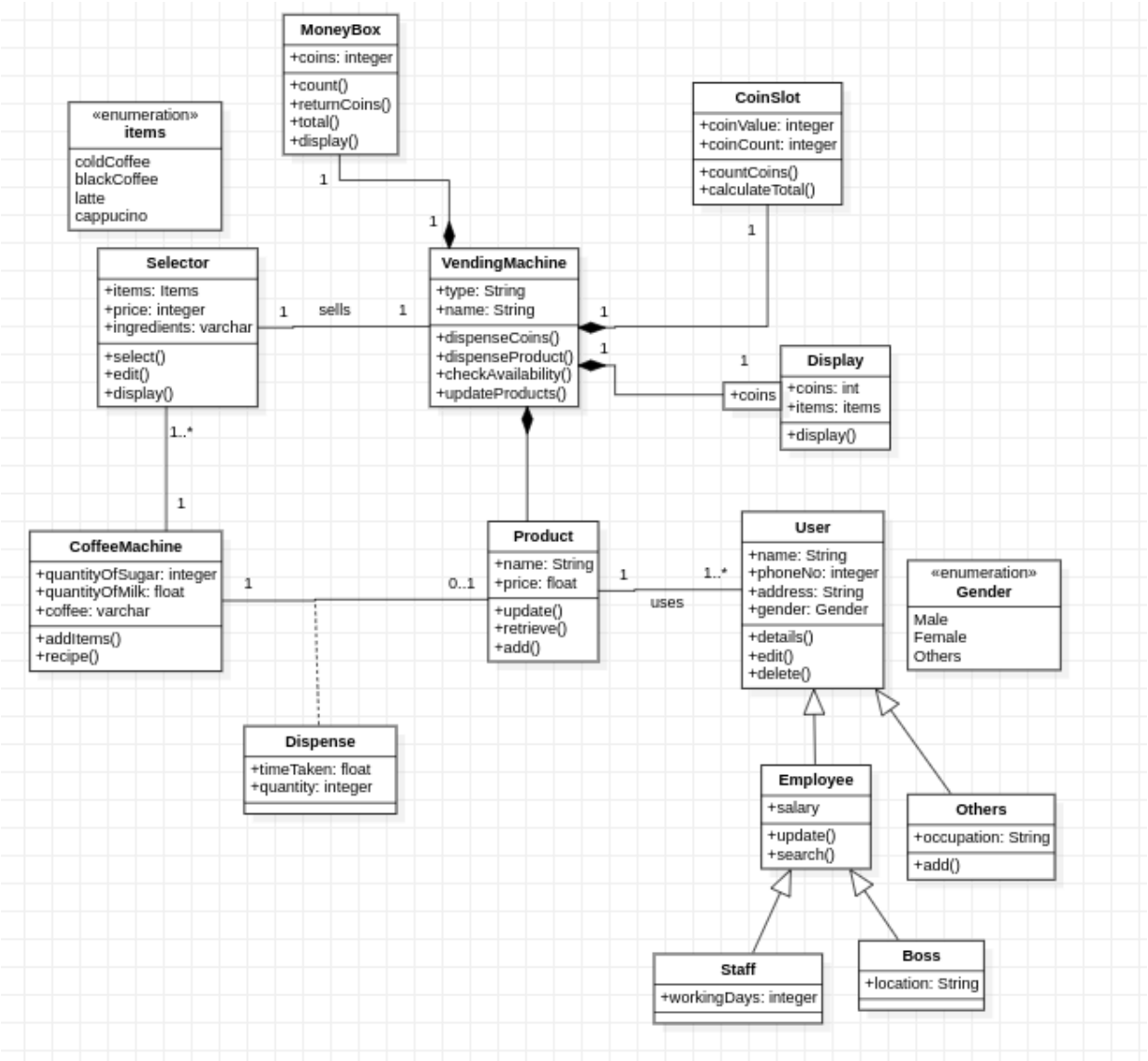
*) The system must accept coins of different amount and the system must compare the item cost with entered coin.

*) The system must check the validity of coins

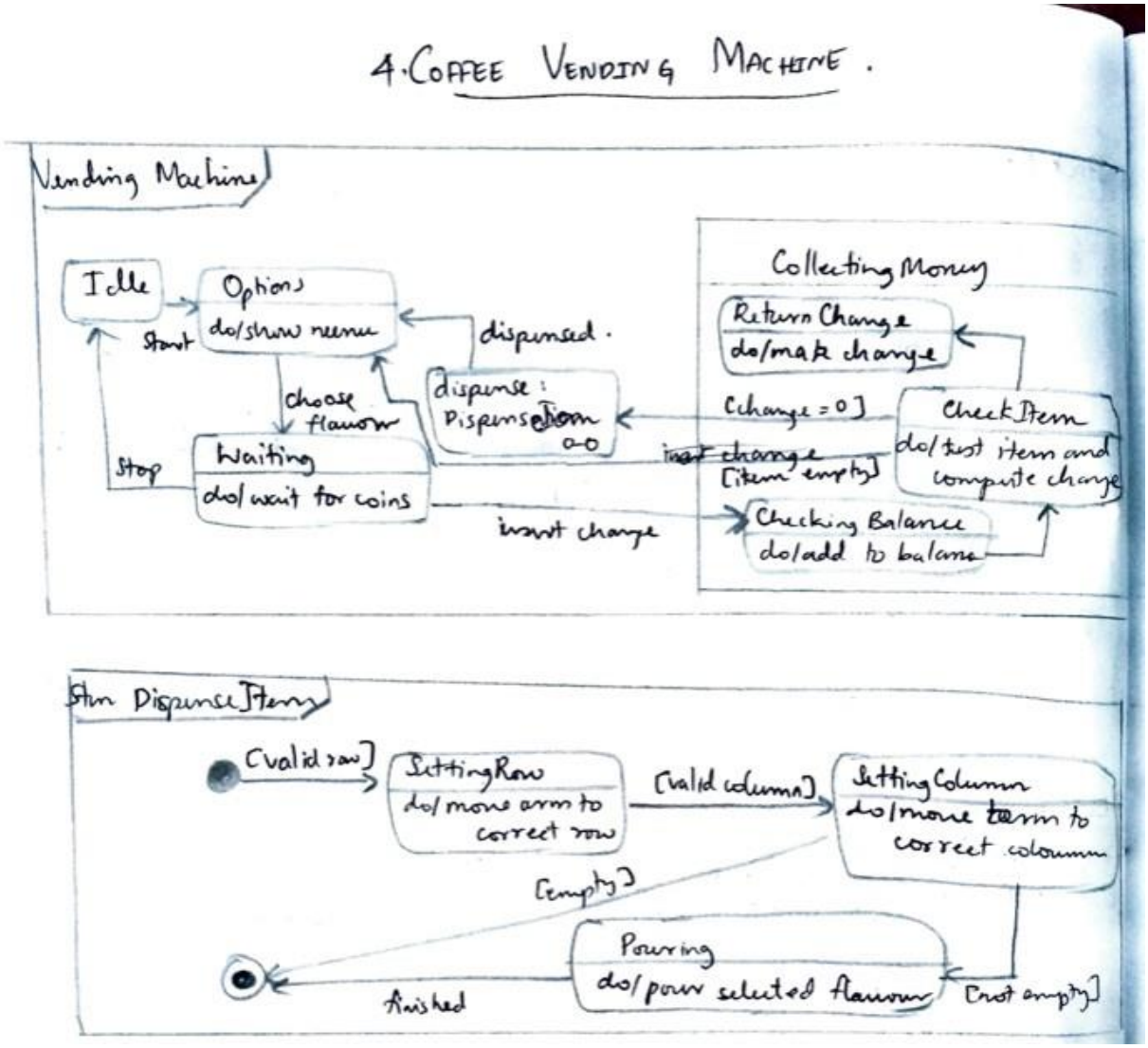
*) The system shall be able to detect the low amount of ingredients and low number of cups and indicate with an indicator.

b) Advance Class Diagram:

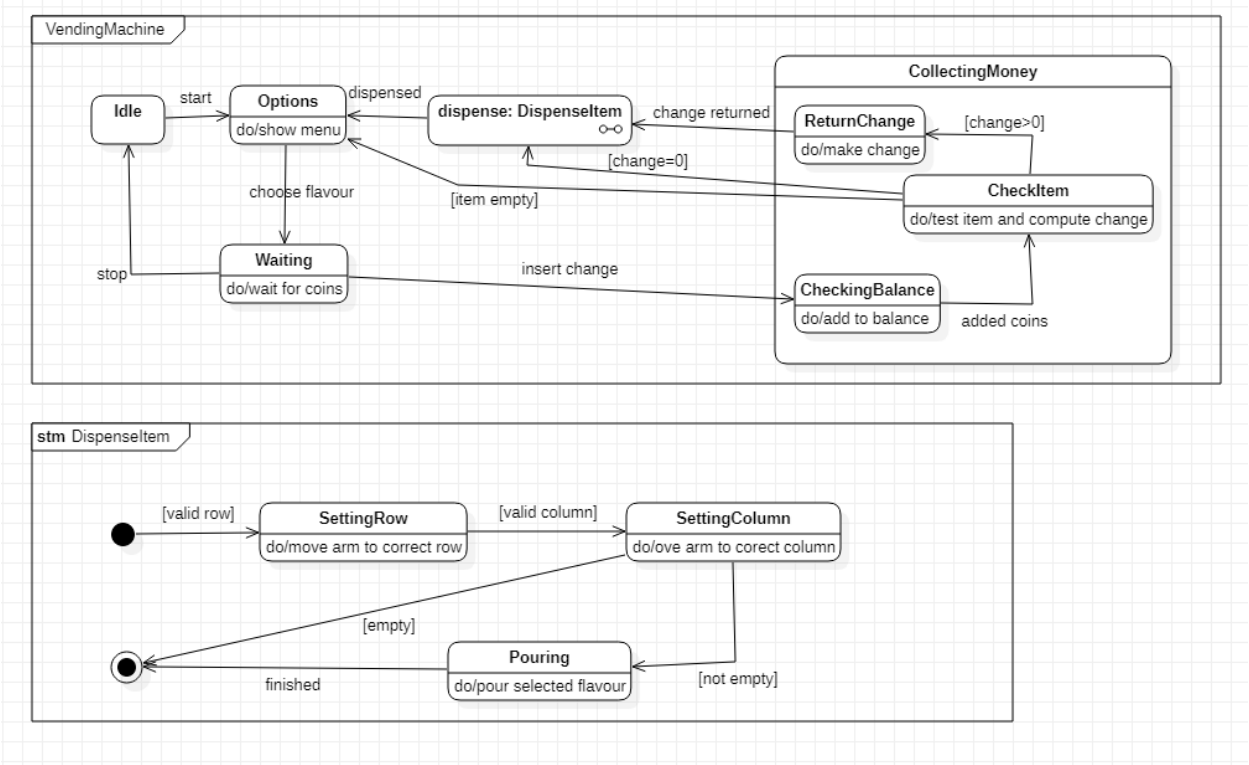
Justification: A coffee vending machine which has coin, product dispenser and coin collector as independent parts and hence they are aggregated, which is composed of product which is in turn composed of coffee, tea and milk. We use association with direction to link to them. Product dispenser is composed of products hence composition.


MoneyBox
+coins: integer
+count()
+returnCoins()
+total()
+display()
«enumeration»
items
coldCoffee
blackCoffee
latte
cappucino
CoinSlot
+coinValue: integer
+coinCount: integer
+countCoins()
+calculateTotal()
Selector
+items: Items
+price: integer
+ingredients: varchar
+select()
+edit()
+display()
VendingMachine
+type: String
+name: String
+dispenseCoins()
+dispenseProduct()
+checkAvailability()
+updateProducts()
Display
+coins: int
+items: items
+display()
+coins
sells
1
1..*
CoffeeMachine
+quantityOfSugar: integer
+quantityOfMilk: float
+coffee: varchar
+addItems()
+recipe()
Product
+name: String
+price: float
+update()
+retrieve()
+add()
0..1
uses
1..*
User
+name: String
+phoneNo: integer
+address: String
+gender: Gender
+details()
+edit()
+delete()
«enumeration»
Gender
Male
Female
Others
Dispense
+timeTaken: float
+quantity: integer
Employee
+salary
+update()
+search()
Others
+occupation: String
+add()
Staff
+workingDays: integer
Boss
+location: String

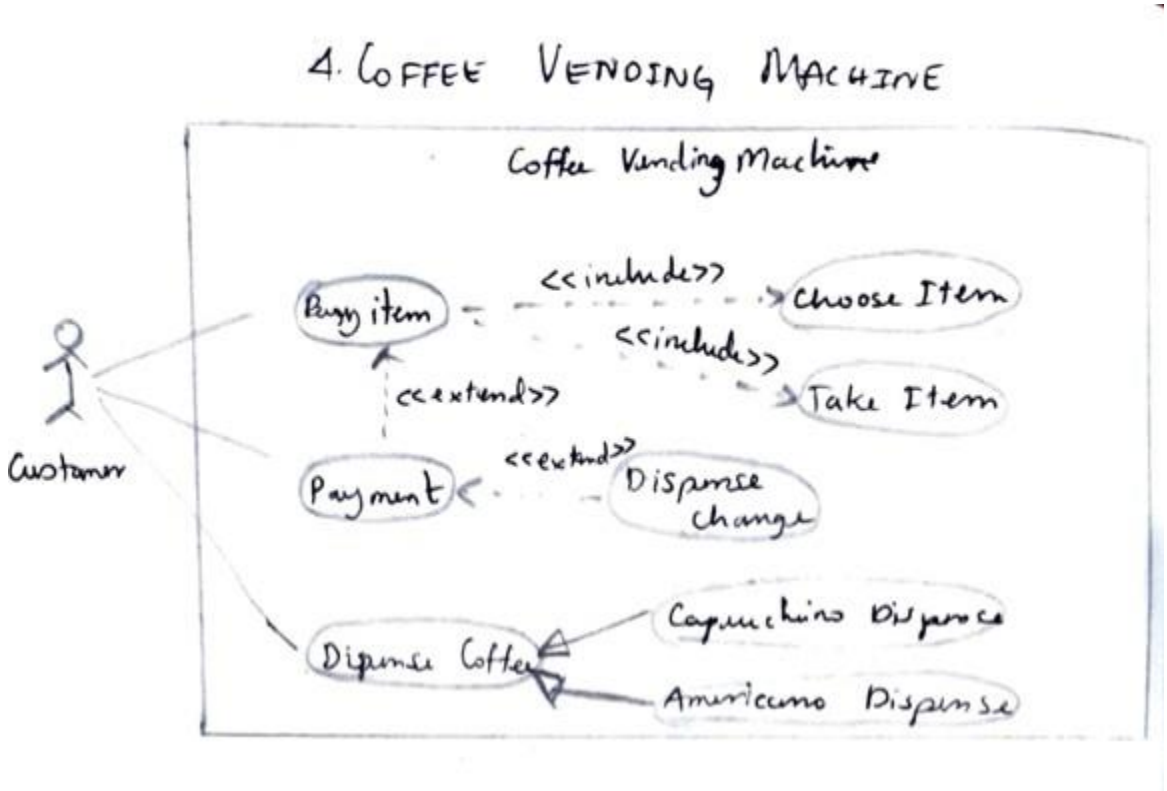
c) Advance State Diagram:



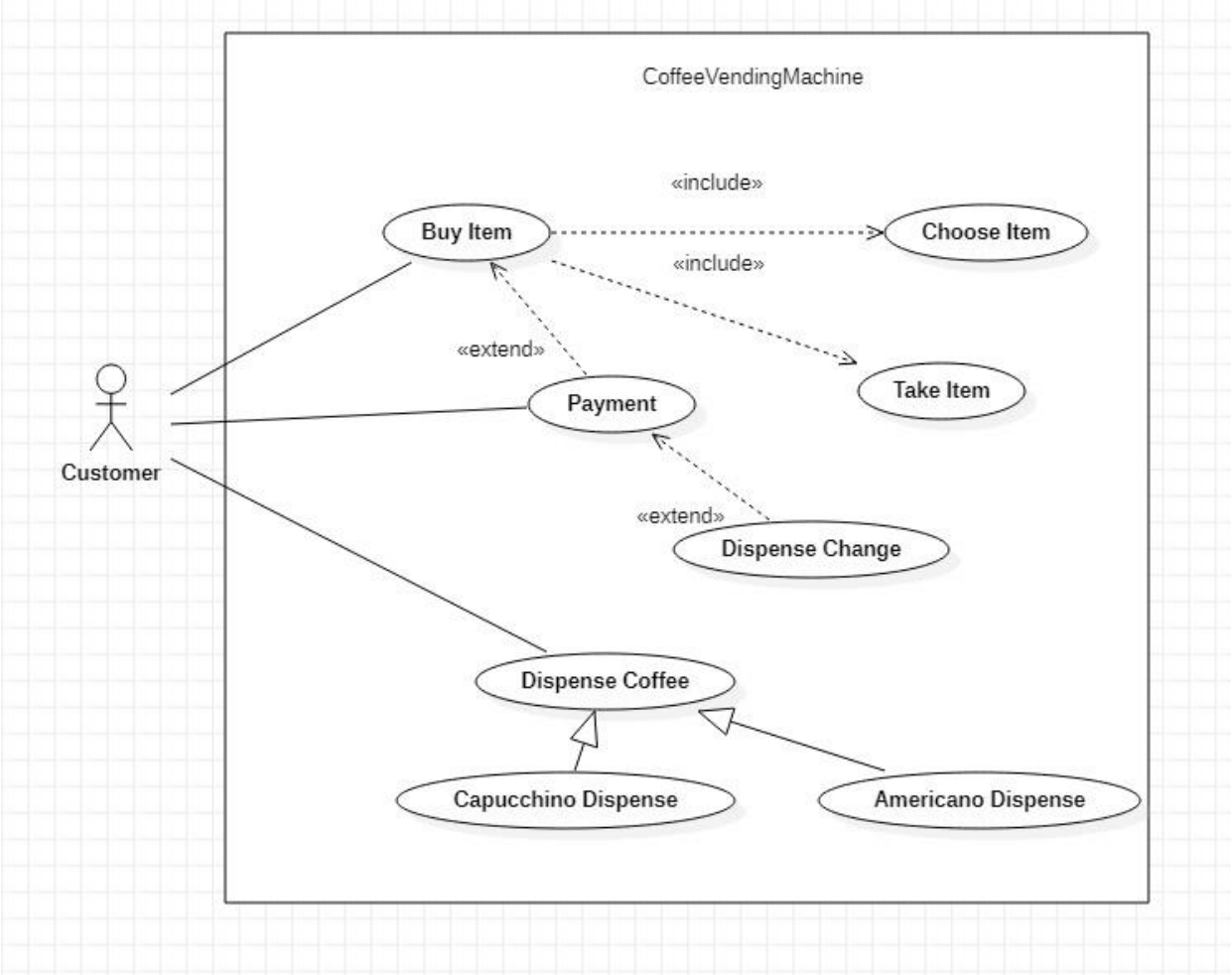
Justification: For the given application, all the respective functions of all the parts are mentioned with all involved actors customer, seller, system and operator.


VendingMachine
Idle
start
Options
do/show menu
dispensed
dispense: DispenseItem
change returned
CollectingMoney
ReturnChange
do/make change
[change>0]
[change=0]
CheckItem
do/test item and compute change
choose flavour
[item empty]
Waiting
do/wait for coins
stop
insert change
CheckingBalance
do/add to balance
added coins
stm DispenseItem
[valid row]
SettingRow
do/move arm to correct row
[valid column]
SettingColumn
do/ove arm to corect column
[empty]
Pouring
do/pour selected flavour
finished
[not empty]

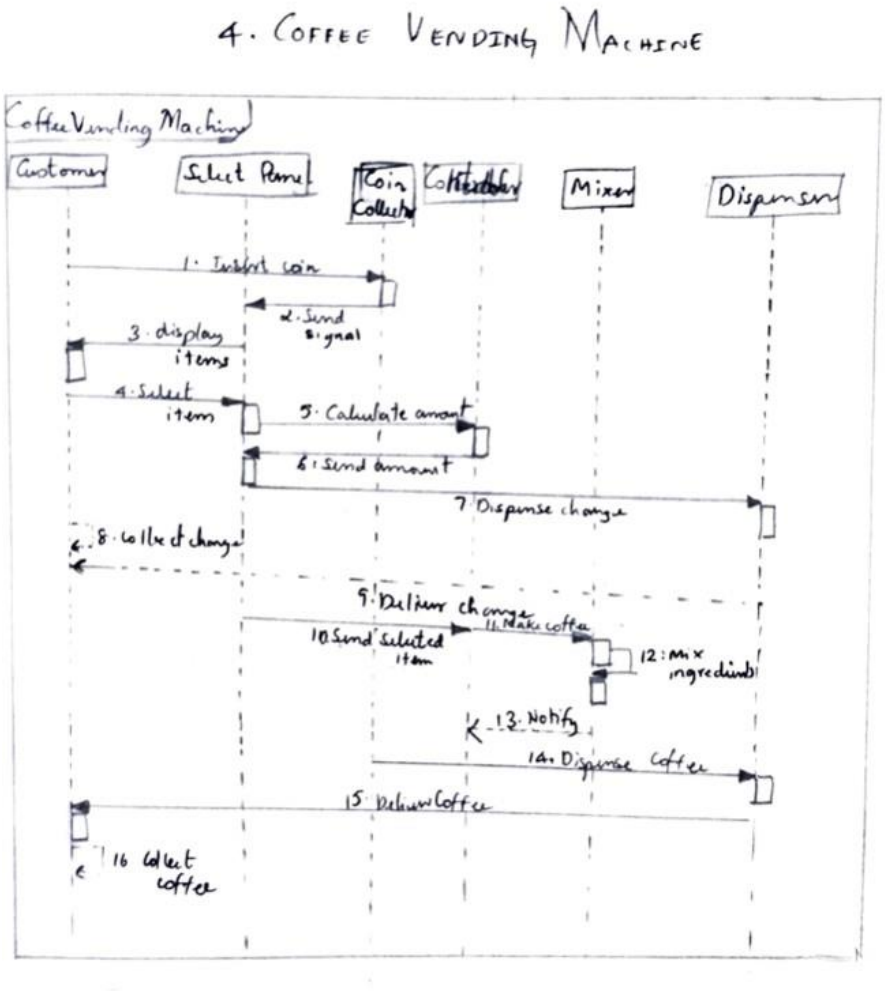
d) Advance Use Case Diagram:



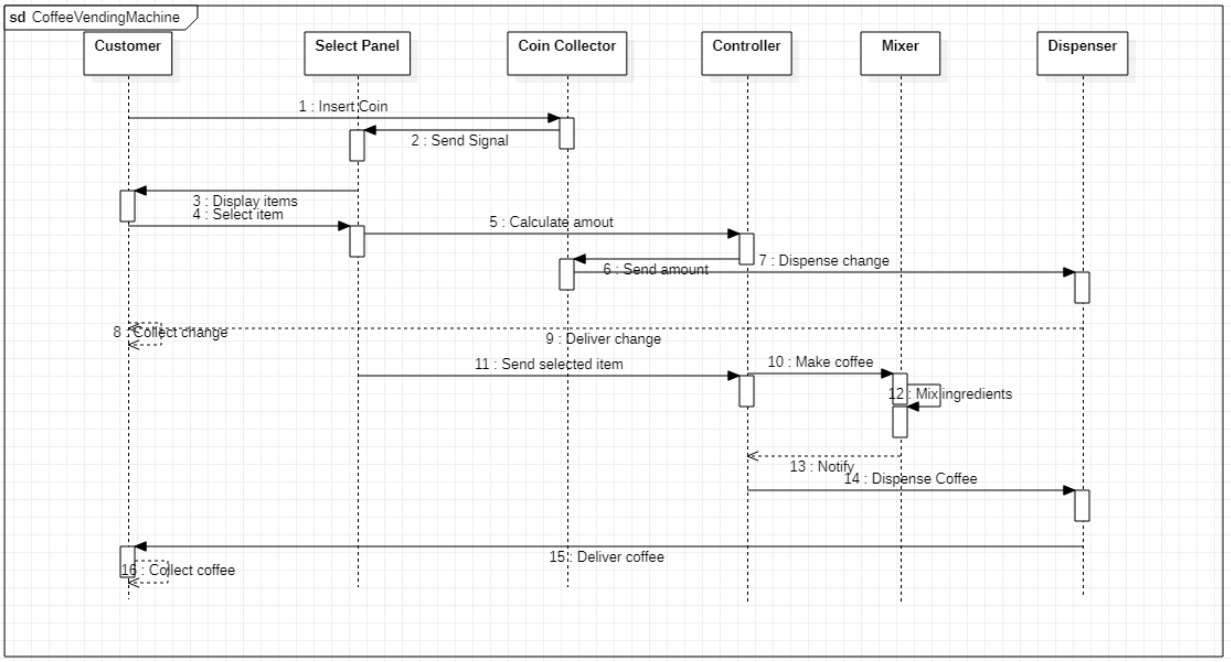
Justification: The given model shows all the interactions of the customer with the coffee vending machine with all respective parts and shows the complete process of how the coffee is dispensed by the system and delivers change.


CoffeeVendingMachine
«include»
Buy Item
Choose Item
«include»
«extend»
Take Item
Payment
Customer
«extend»
Dispense Change
Dispense Coffee
Capucchino Dispense
Americano Dispense

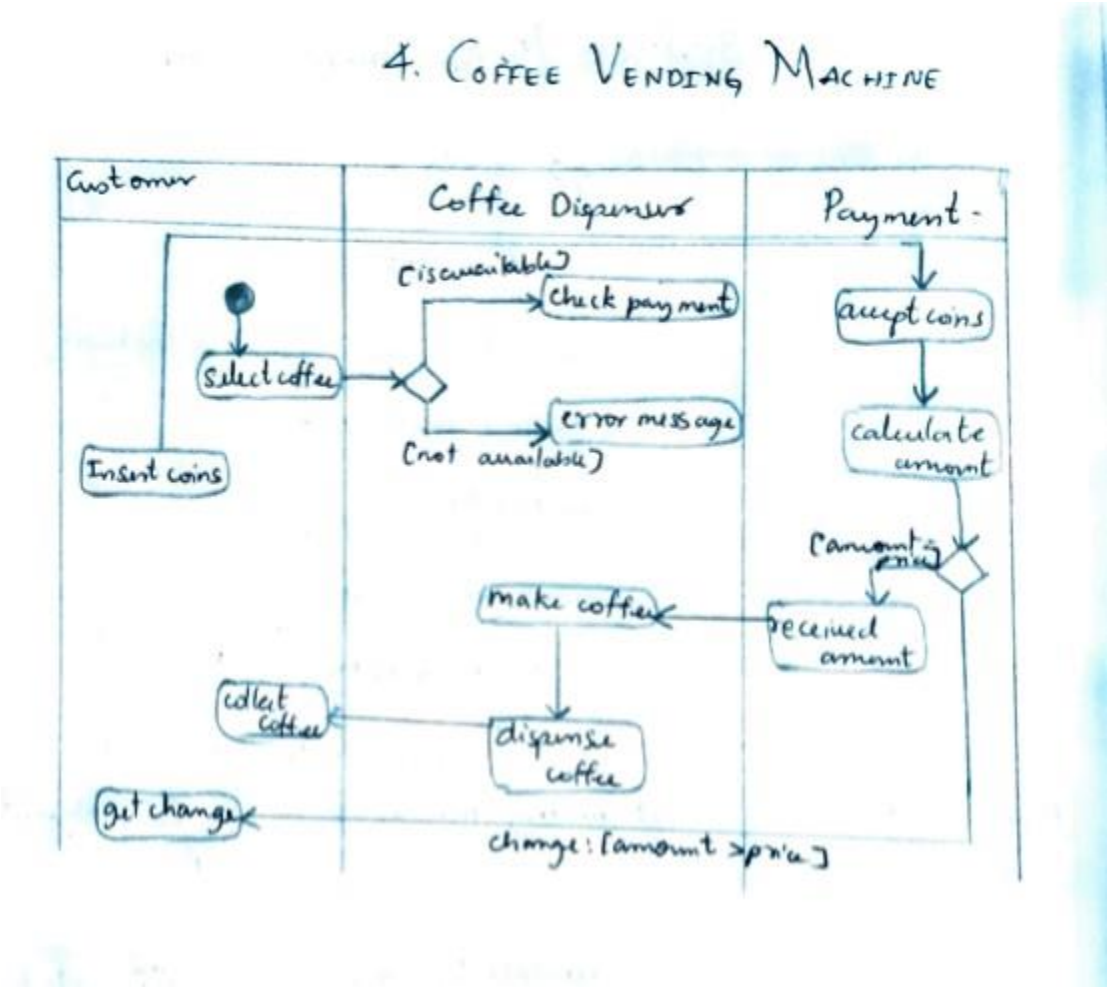
e) Sequence Diagram:



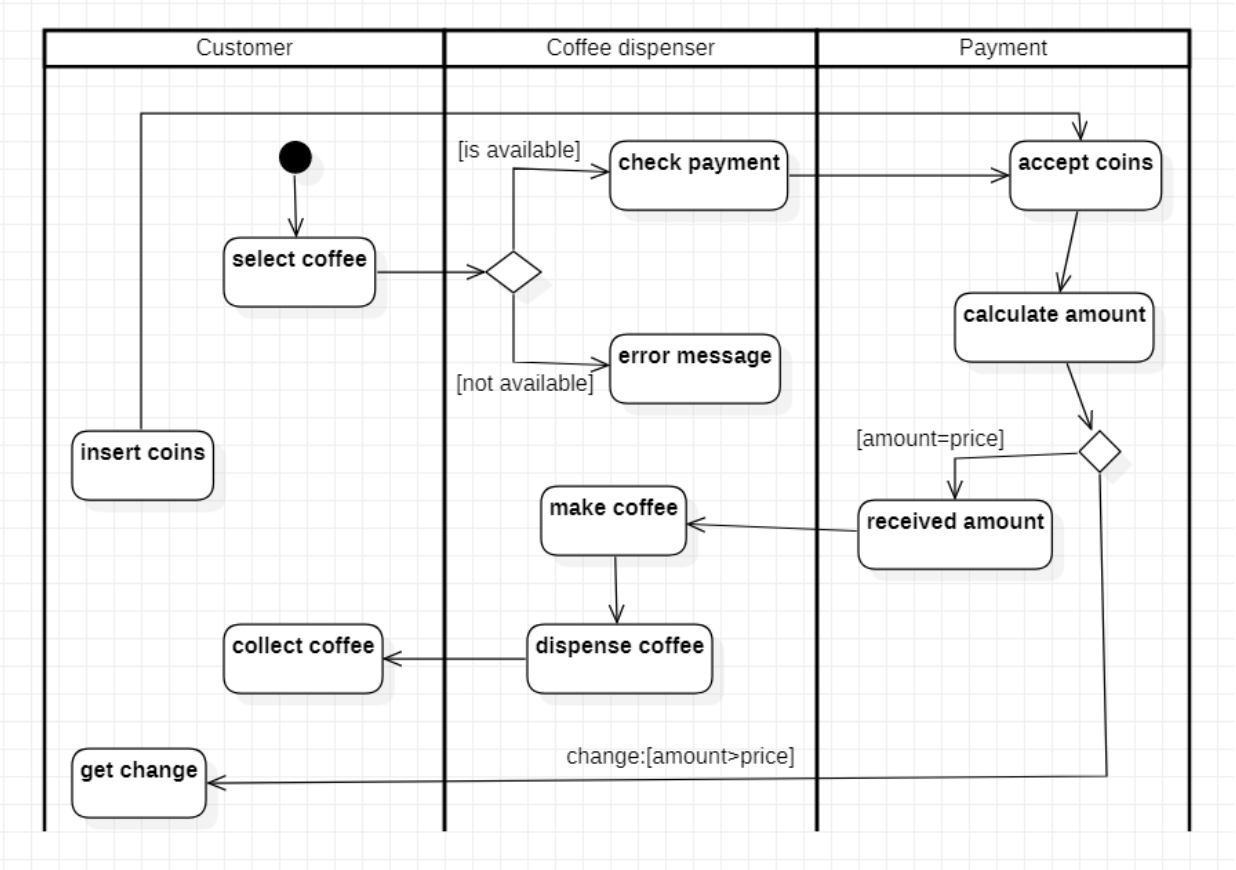
Justification: The given sequence diagram shows the appropriate sequence of processes with respect to the given system. It shows how user enters input and system responds appropriately.


sd CoffeeVendingMachine
Customer
Select Panel
Coin Collector
Controller
Mixer
Dispenser
1 : Insert Coin
2 : Send Signal
3 : Display items
4 : Select item
5 : Calculate amout
6 : Send amount
7 : Dispense change
8 : Collect change
9 : Deliver change
11 : Send selected item
10 : Make coffee
12 : Mixingredients
13 : Notify
14 : Dispense Coffee
15 : Deliver coffee
16 : Collect coffee

f) Activity Diagram:

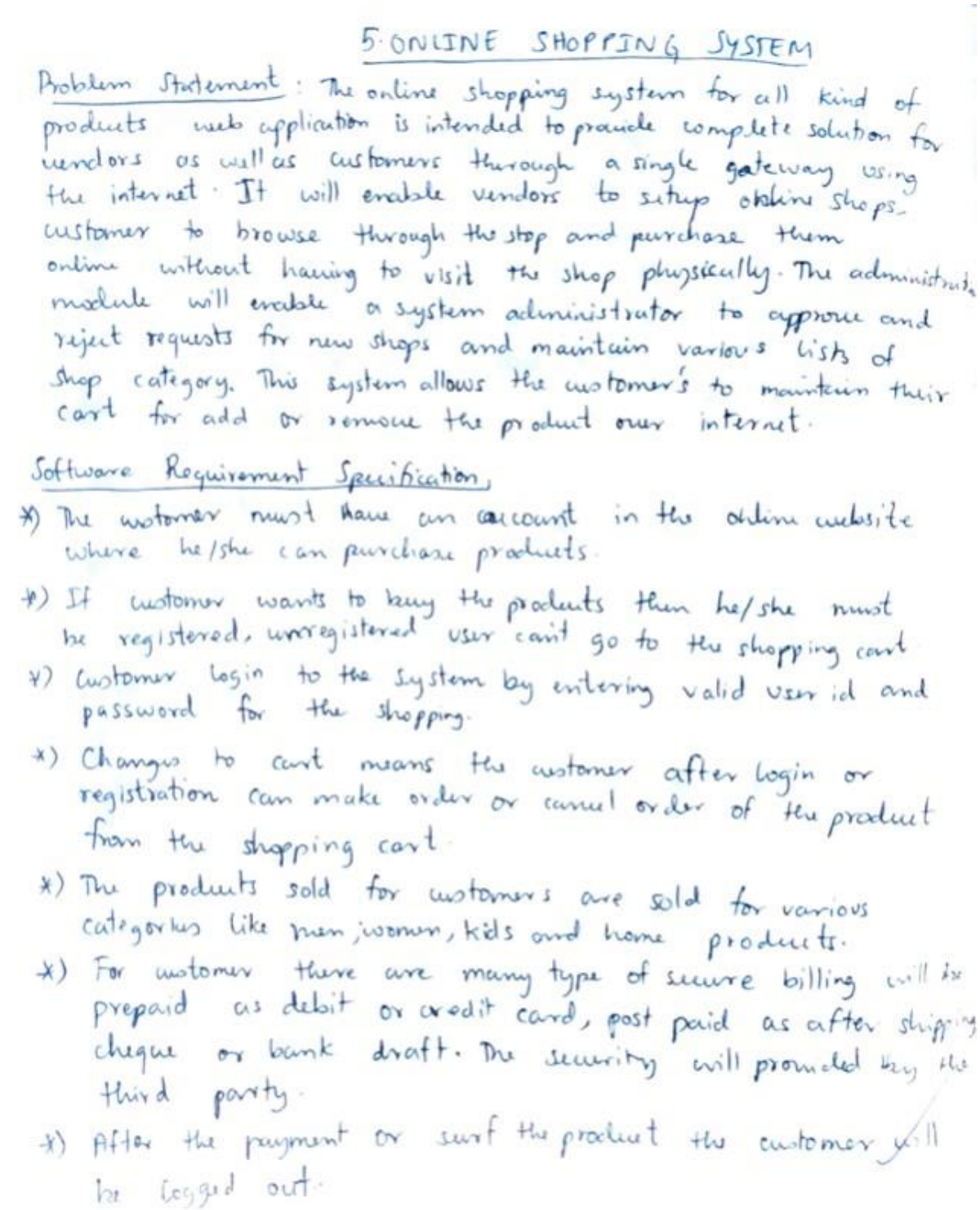


Justification: The given model explains the complete interaction between customer and the system from selecting the type of coffee, entering money into the machine and dispensation of the same.


Customer
Coffee dispenser
Payment
[is available]
check payment
accept coins
select coffee
calculate amount
error message
[not available]
insert coins
[amount=price]
make coffee
received amount
collect coffee
dispense coffee
change:[amount>price]
get change

5. Online Shopping System-

a) SRS:



5. ONLINE SHOPPING SYSTEM

Problem Statement : The online shopping system for all kind of products web application is intended to provide complete solution for vendors as well as customers through a single gateway using the internet. It will enable vendors to setup online shops, customer to browse through the stop and purchase them online without having to visit the shop physically. The administrator module will enable a system administrator to approve and reject requests for new shops and maintain various lists of shop category. This system allows the customer's to maintain their cart for add or remove the product over internet.

Software Requirement Specification,

*) The customer must have an account in the online website where he/she can purchase products.

*) If customer wants to buy the products then he/she must be registered, unregistered user can't go to the shopping cart.

*) Customer login to the system by entering valid user id and password for the shopping.

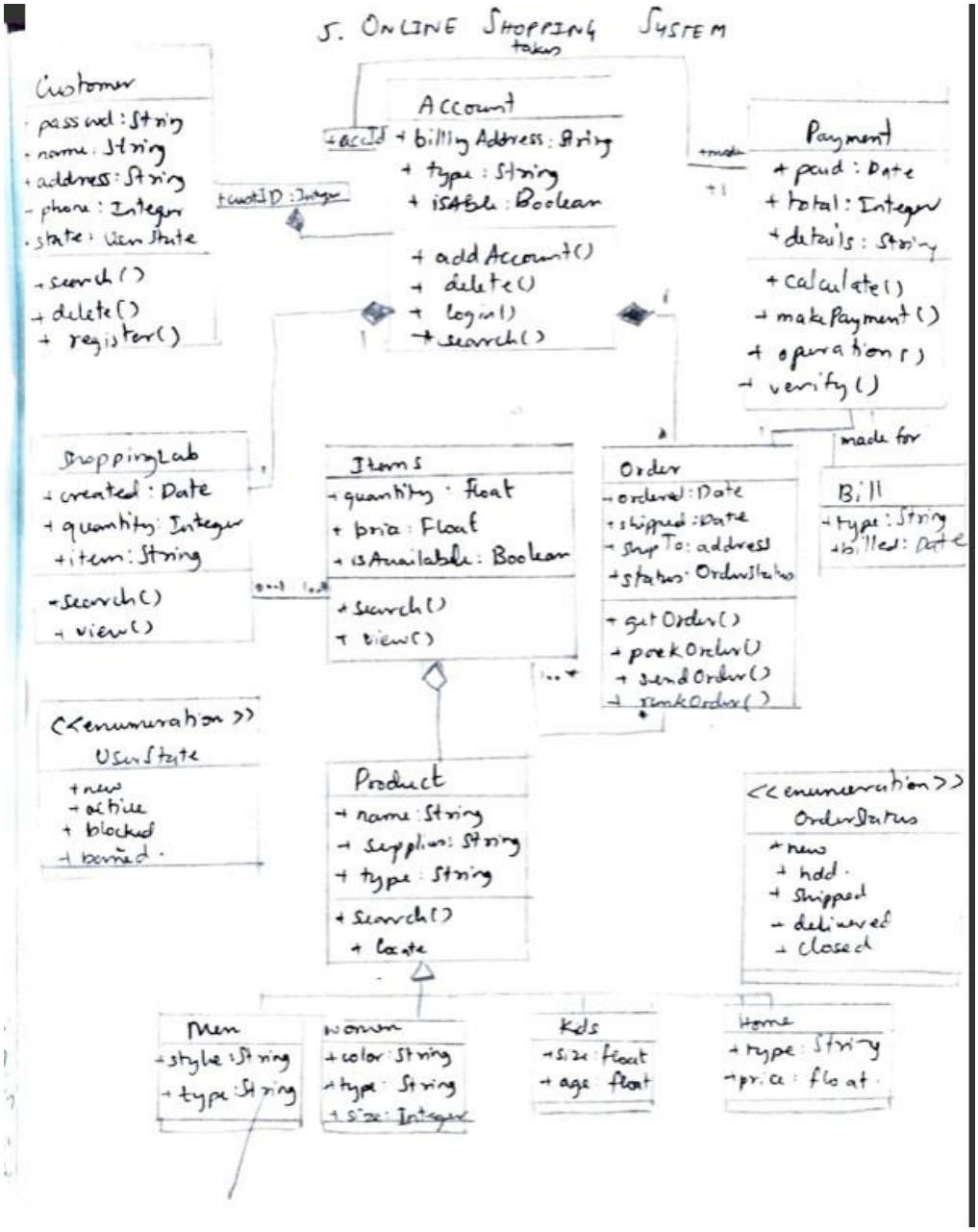
*) Changes to cart means the customer after login or registration can make order or cancel order of the product from the shopping cart.

*) The products sold for customers are sold for various categories like men, women, kids and home products.

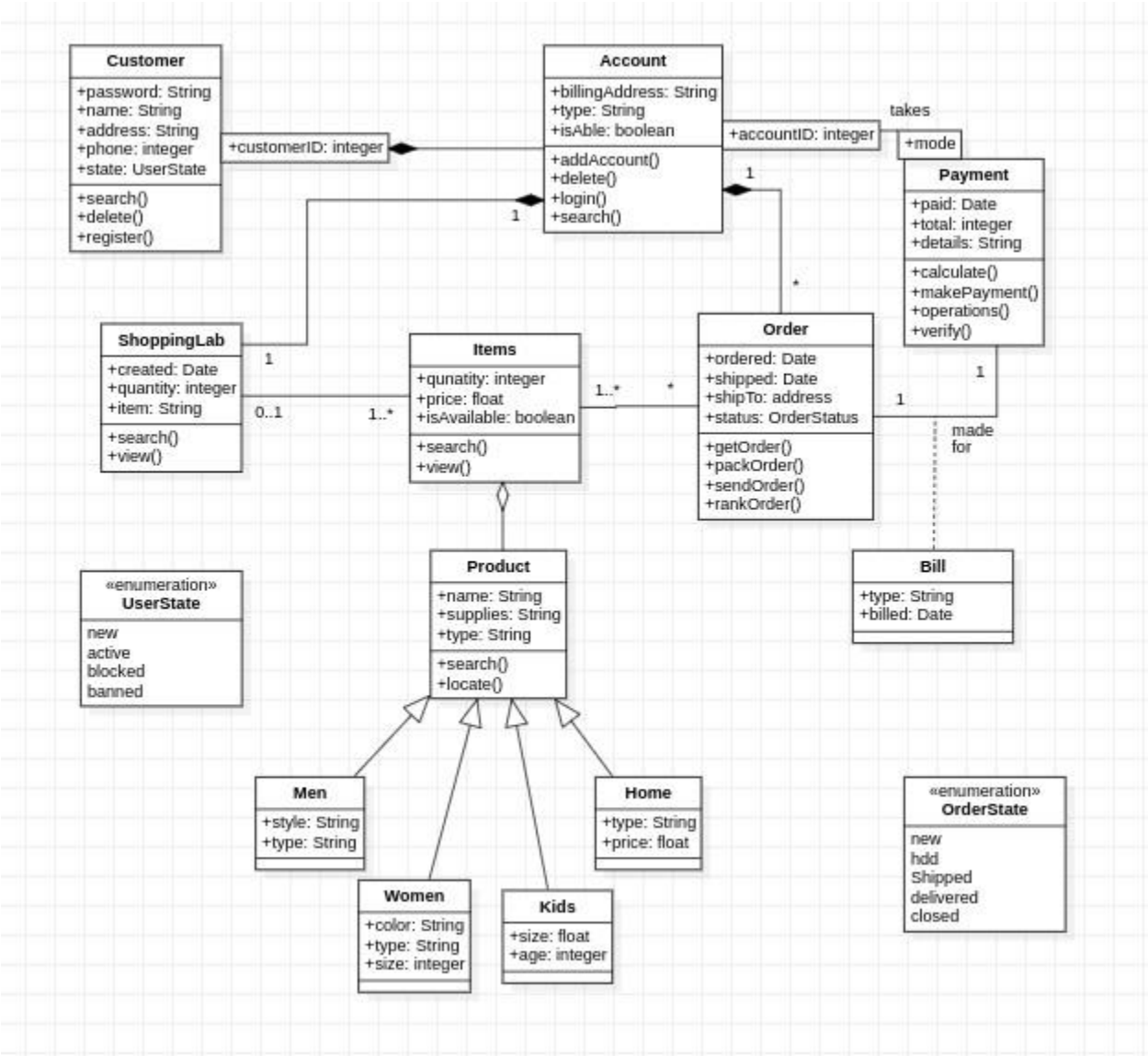
*) For customer there are many type of secure billing will be prepaid as debit or credit card, post paid as after shipping cheque or bank draft. The security will provided by the third party.

*) After the payment or surf the product the customer will be logged out.

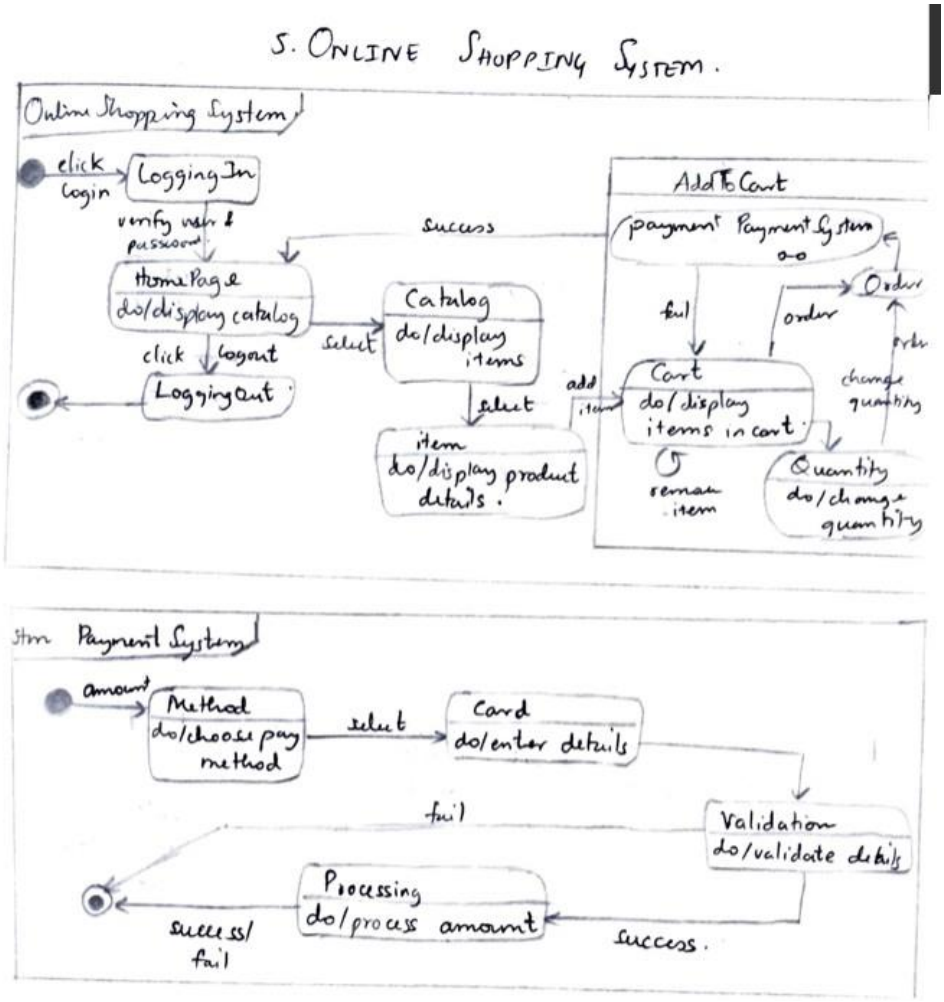
b) Advance Class Diagram:



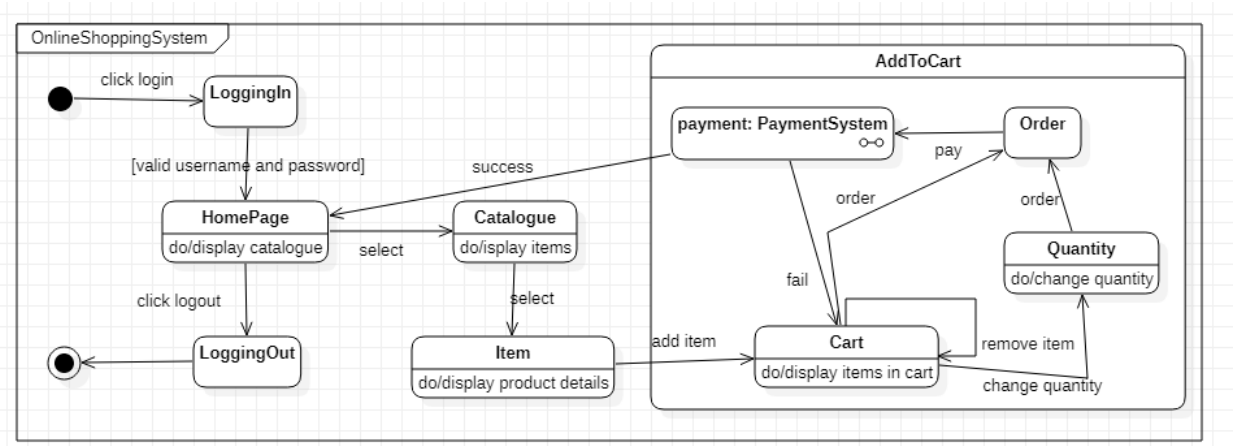
Justification: Every customer is linked with an account using id, where the account cannot exist without the customer, so it is composed. Every account has a shopping cart and order is associated with items which are placed in the shopping cart. Items are aggregated with products which are generalized into various classifications. The account, payment for the order to the bill is linked by association.

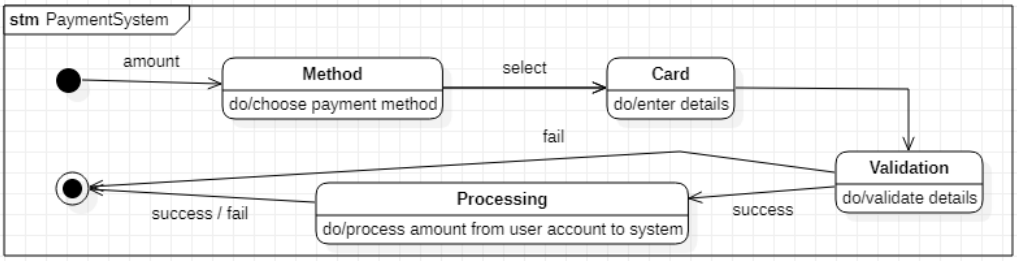

Customer
+password: String
+name: String
+address: String
+phone: integer
+state: UserState
+search()
+delete()
+register()
+customerID: integer
Account
+billingAddress: String
+type: String
+isAble: boolean
+addAccount()
+delete()
+login()
+search()
+accountID: integer
takes
+mode
Payment
+paid: Date
+total: integer
+details: String
+calculate()
+makePayment()
+operations()
+verify()
1
1
*
1
made
for
ShoppingLab
+created: Date
+quantity: integer
+item: String
+search()
+view()
1
0..1
1..*
Items
+qunatity: integer
+price: float
+isAvailable: boolean
+search()
+view()
1..*
*
Order
+ordered: Date
+shipped: Date
+shipTo: address
+status: OrderStatus
+getOrder()
+packOrder()
+sendOrder()
+rankOrder()
«enumeration»
UserState
new
active
blocked
banned
Product
+name: String
+supplies: String
+type: String
+search()
+locate()
Bill
+type: String
+billed: Date
Men
+style: String
+type: String
Women
+color: String
+type: String
+size: integer
Kids
+size: float
+age: integer
Home
+type: String
+price: float
«enumeration»
OrderState
new
hdd
Shipped
delivered
closed

c) Advance State Diagram:

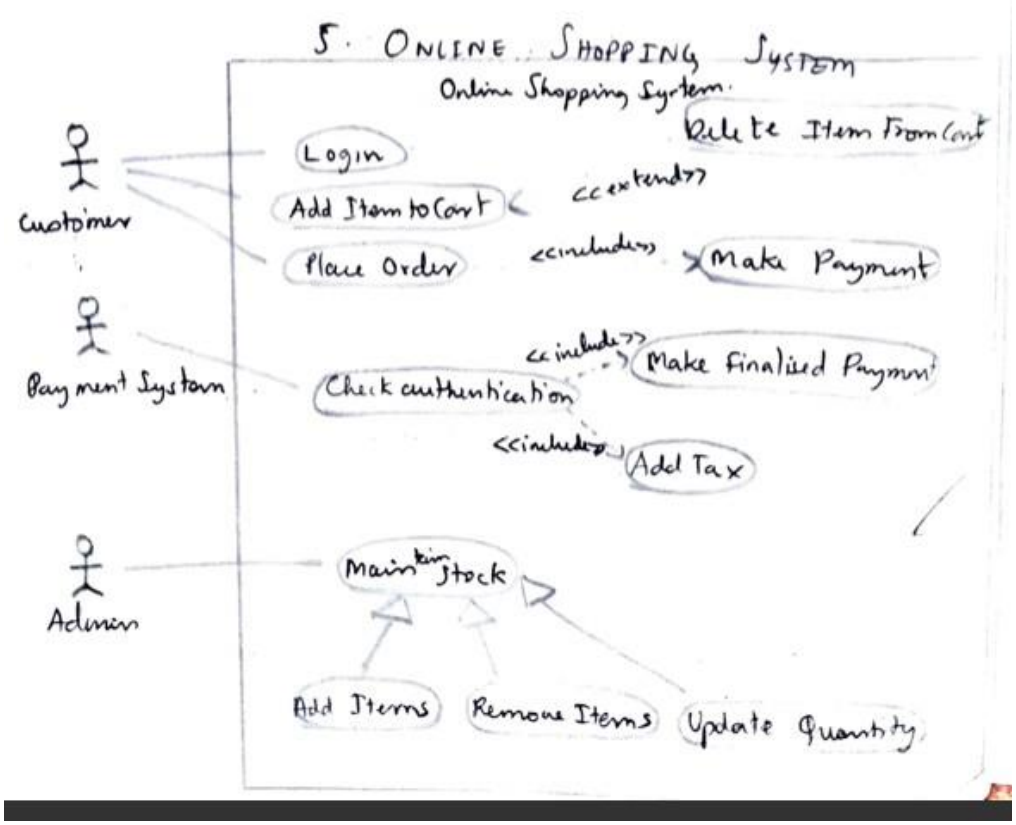


Justification: The given model explains with respect to an add to cart and payment system scenario with proper transactions and actions. The detailed composition and submachine model with the proper details.

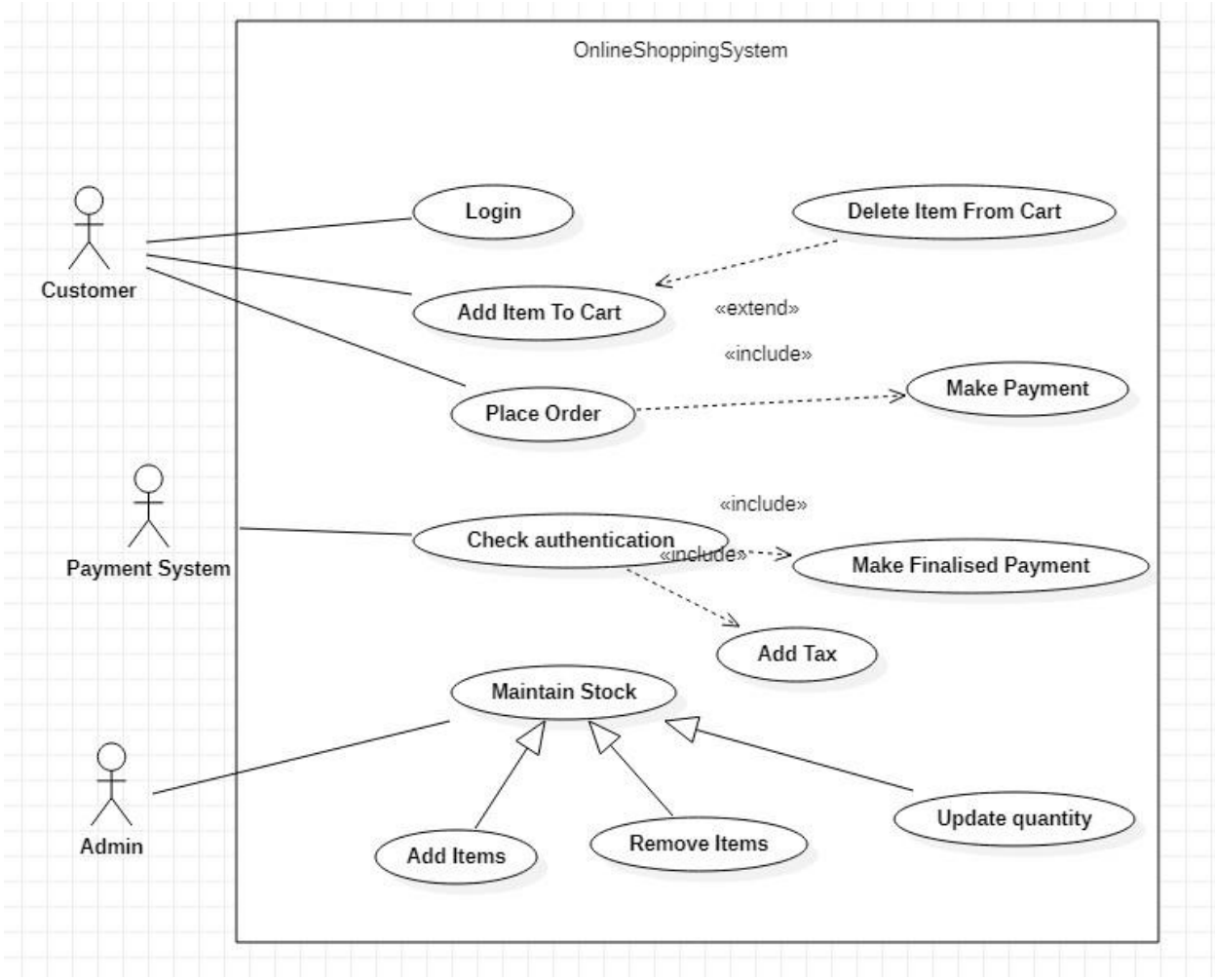

OnlineShoppingSystem
click login
LoggingIn
[valid username and password]
HomePage
do/display catalogue
select
Catalogue
do/isplay items
click logout
LoggingOut
select
Item
do/display product details
AddToCart
payment: PaymentSystem
success
Order
pay
order
order
Quantity
do/change quantity
fail
add item
Cart
do/display items in cart
remove item
change quantity


stm PaymentSystem
amount
Method
do/choose payment method
select
Card
do/enter details
fail
Validation
do/validate details
success / fail
Processing
do/process amount from user account to system
success

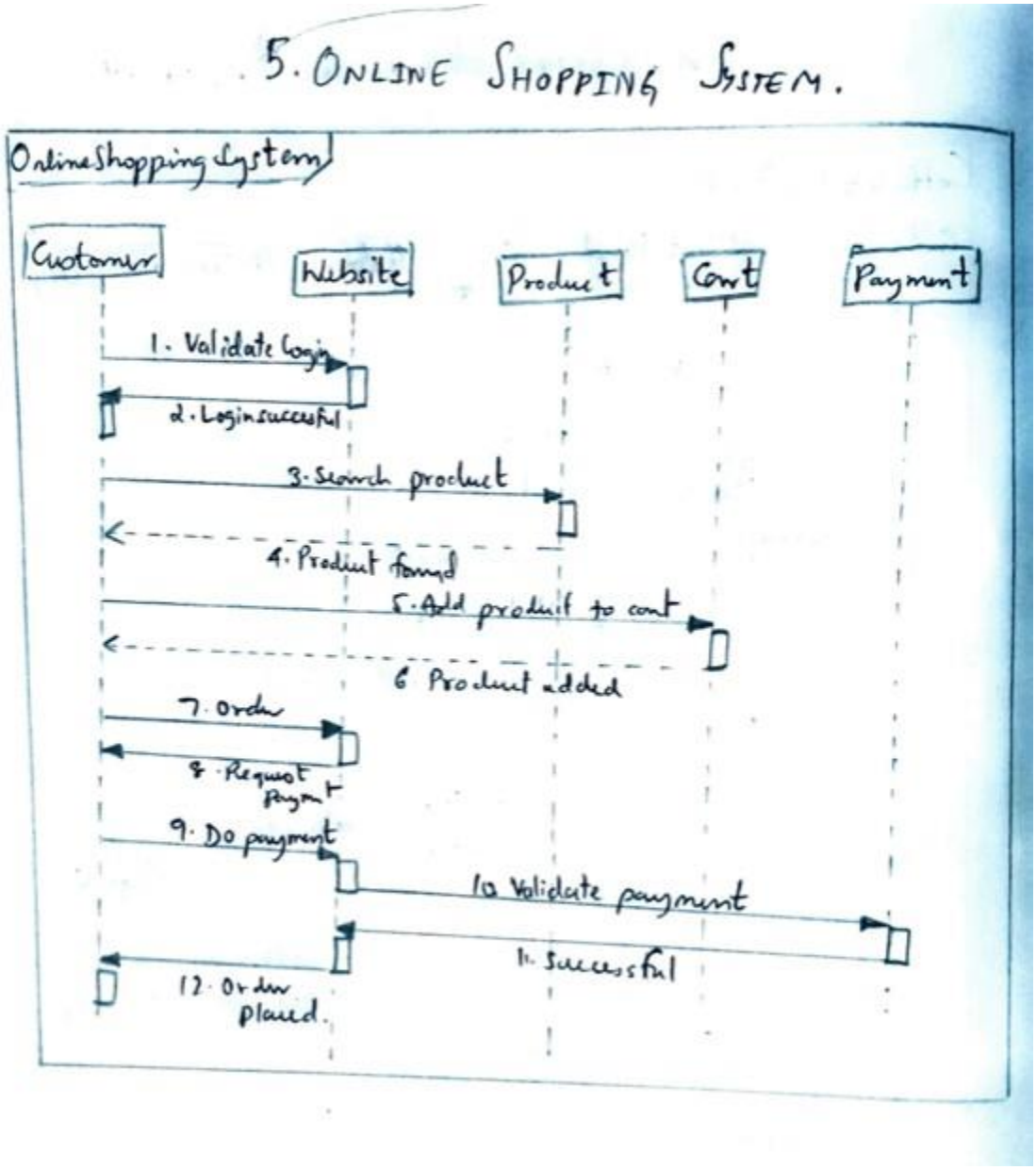
d) Advance Use Case Diagram:



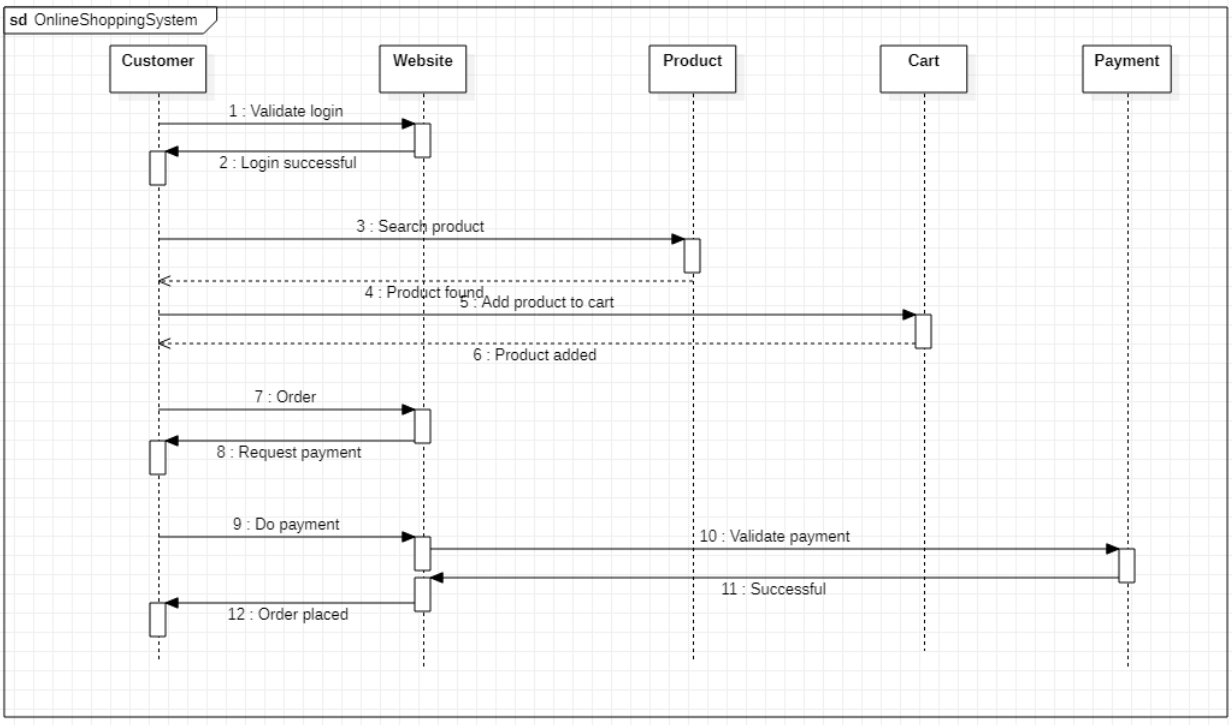
Justification: All the functions for the given system are given properly with all functions of the applications and all the actors are like customer, admin, payment system are given and all the relations for the given working is mentioned clearly.


OnlineShoppingSystem
Customer
Login
Delete Item From Cart
Add Item To Cart
«extend»
«include»
Place Order
Make Payment
Payment System
«include»
Check authentication
«include»
Make Finalised Payment
Add Tax
Maintain Stock
Admin
Add Items
Remove Items
Update quantity

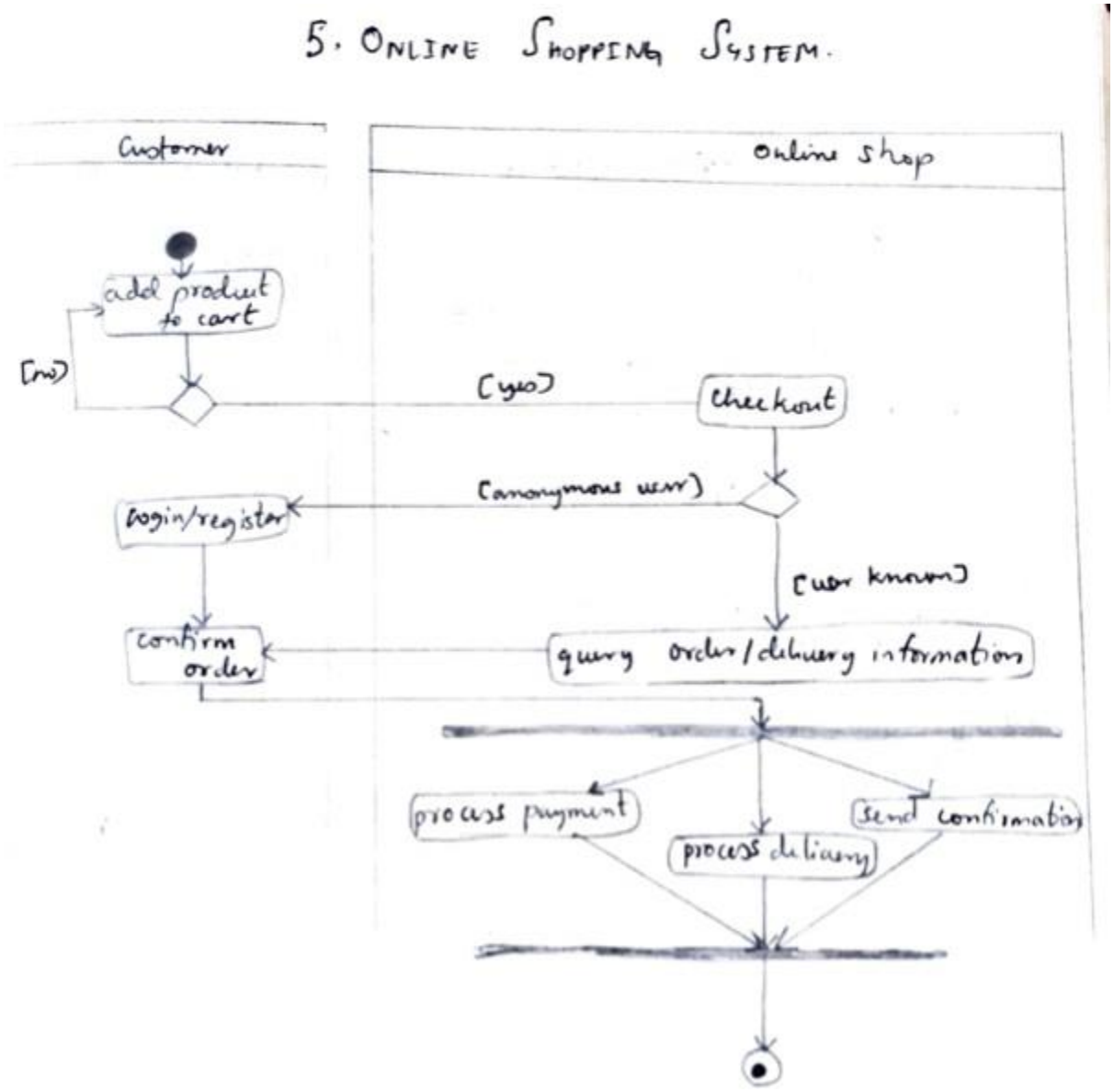
e) Sequence Diagram:



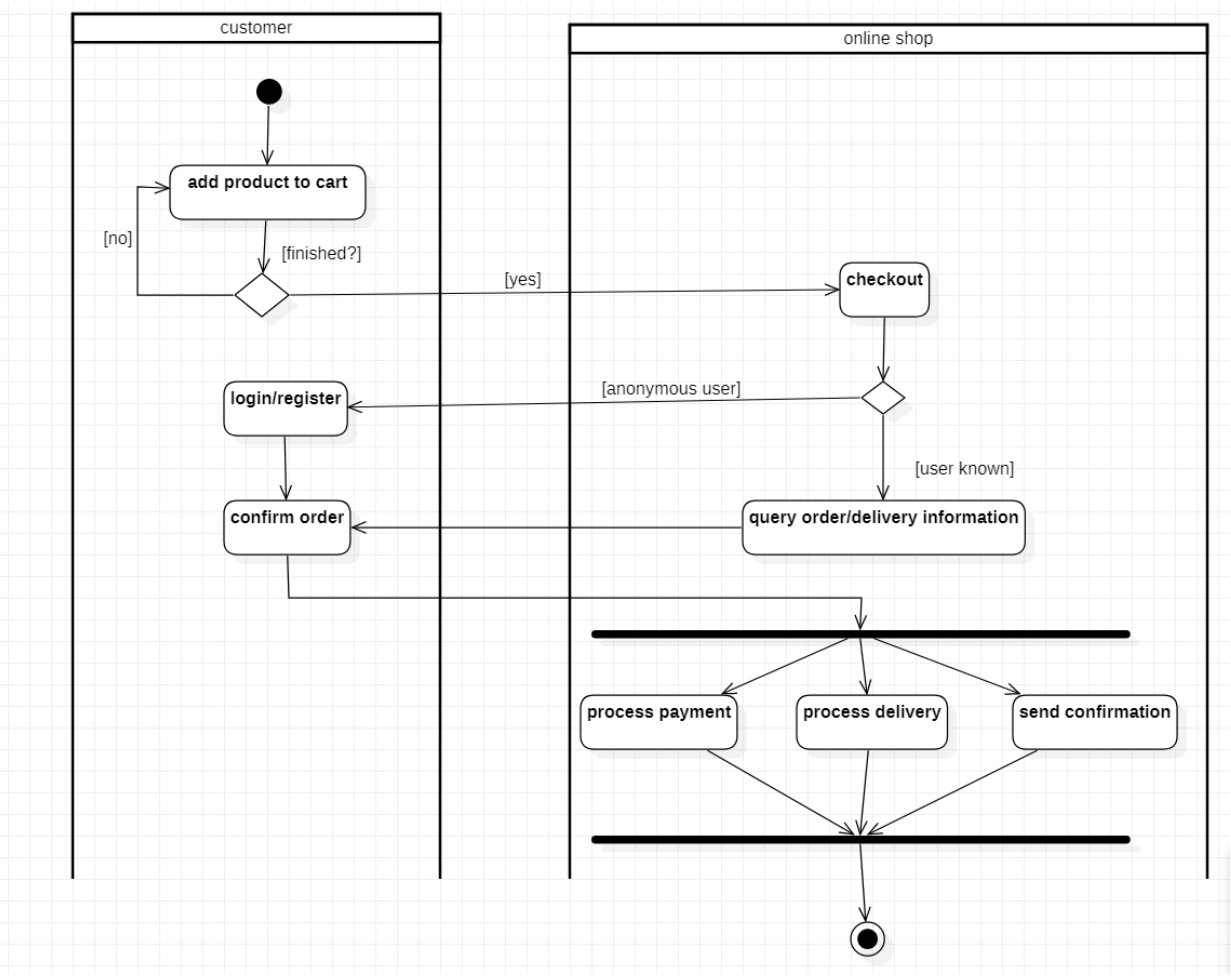
Justification: The given system shows the complete interaction between of the customer in the process of purchasing the products from searching to booking to payment and delivery and all the respective activities are mentioned.


sd OnlineShoppingSystem
Customer
Website
Product
Cart
Payment
1 : Validate login
2 : Login successful
3 : Search product
4 : Product found
5 : Add product to cart
6 : Product added
7 : Order
8 : Request payment
9 : Do payment
10 : Validate payment
11 : Successful
12 : Order placed

f) Activity Diagram:

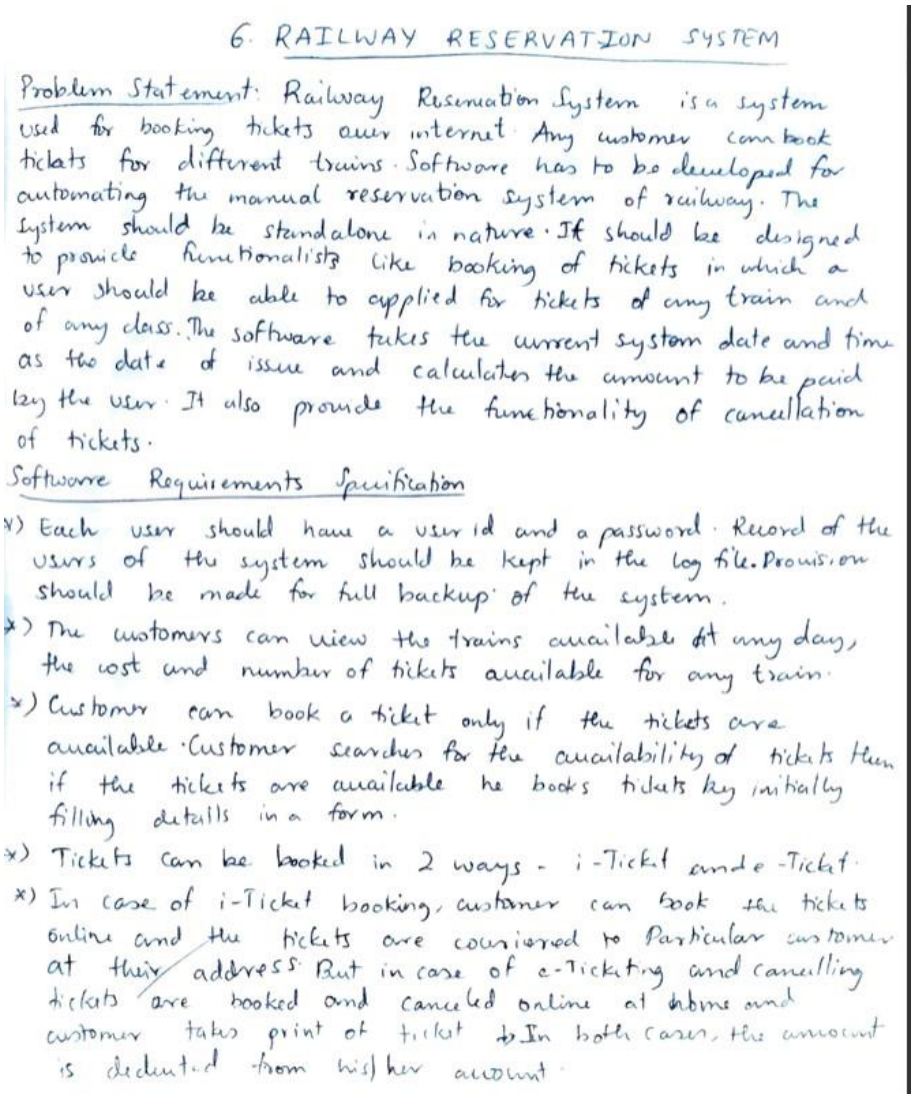


Justification: The given activity model explains the various interactions between taking place during the given customer login and shops from add to cart checkout and payment to process delivery all the given respective interactions are mentioned.


customer
online shop
add product to cart
[no]
[finished?]
[yes]
checkout
[anonymous user]
login/register
[user known]
confirm order
query order/delivery information
process payment
process delivery
send confirmation

6. Railway reservation system-

a) SRS:



6. RAILWAY RESERVATION SYSTEM

Problem Statement: Railway Reservation System is a system used for booking tickets over internet. Any customer can book tickets for different trains. Software has to be developed for automating the manual reservation system of railway. The system should be standalone in nature. It should be designed to provide functionalities like booking of tickets in which a user should be able to applied for tickets of any train and of any class. The software takes the current system date and time as the date of issue and calculates the amount to be paid by the user. It also provide the functionality of cancellation of tickets.

Software Requirements Specification

*) Each user should have a user id and a password. Record of the users of the system should be kept in the log file. Provision should be made for full backup of the system.

*) The customers can view the trains available at any day, the cost and number of tickets available for any train.

*) Customer can book a ticket only if the tickets are available. Customer searches for the availability of tickets then if the tickets are available he books tickets by initially filling details in a form.

*) Tickets can be booked in 2 ways - i-Ticket and e-Ticket.

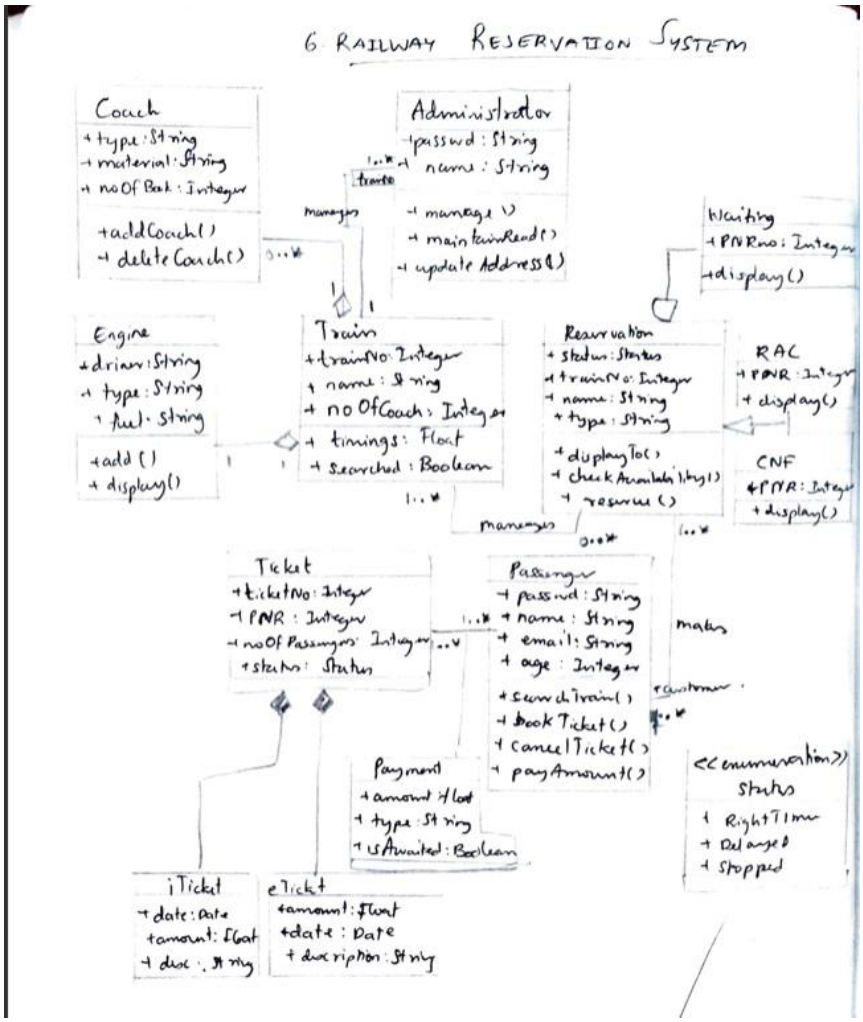
*) In case of i-Ticket booking, customer can book the tickets online and the tickets are couriered to Particular customer at their address. But in case of e-Ticketing and cancelling tickets are booked and canceled online at home and customer takes print of ticket. In both cases, the amount is deducted from his/her account.

*) For cancellation of ticket, the customer has to go at reservation office then fill cancellation form and ask the clerk to cancel the ticket. Then the refund is transferred to customer account.

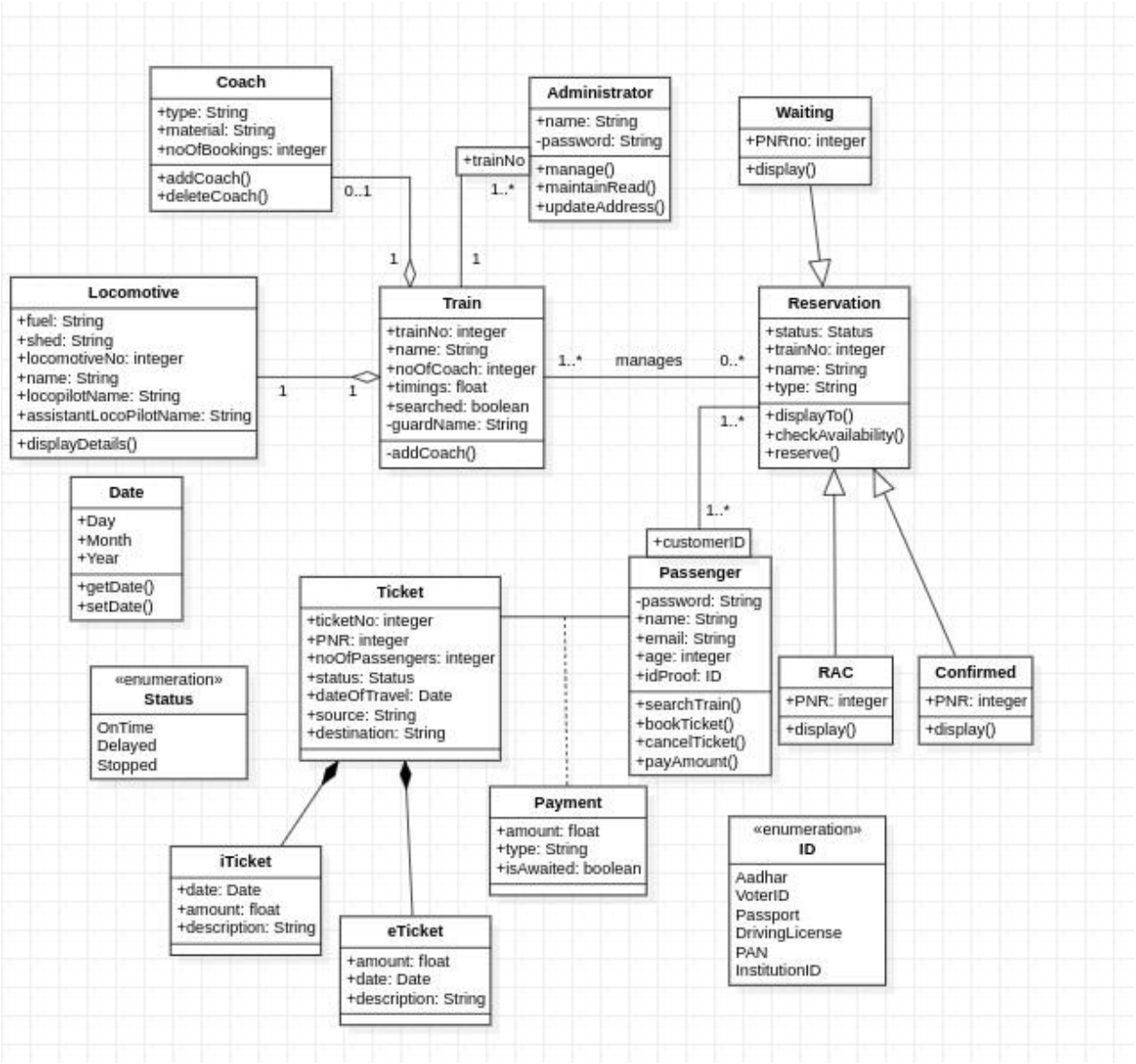
*) After booking ticket the customer has to checkout by paying fare amount to clerk.

*) The system displays the details of train of which user enter the name. The information is saved and the corresponding updating take place in the database.

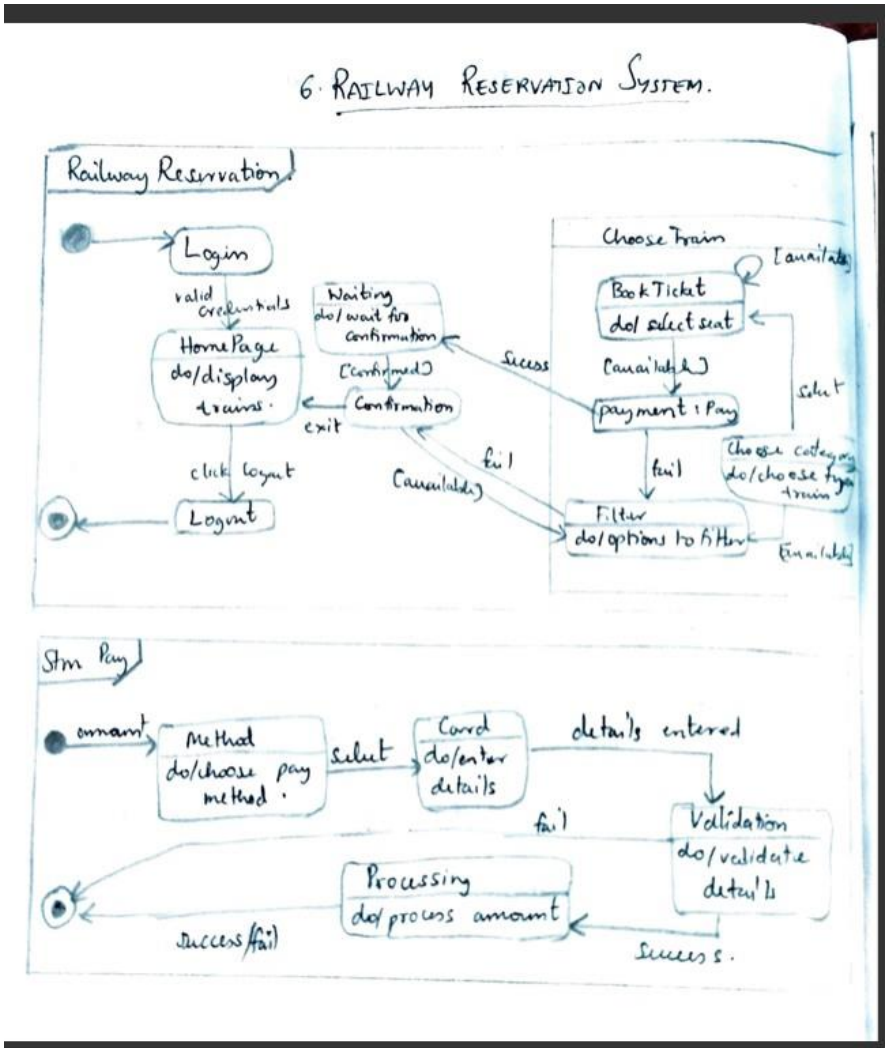
b) Advance Class Diagram:



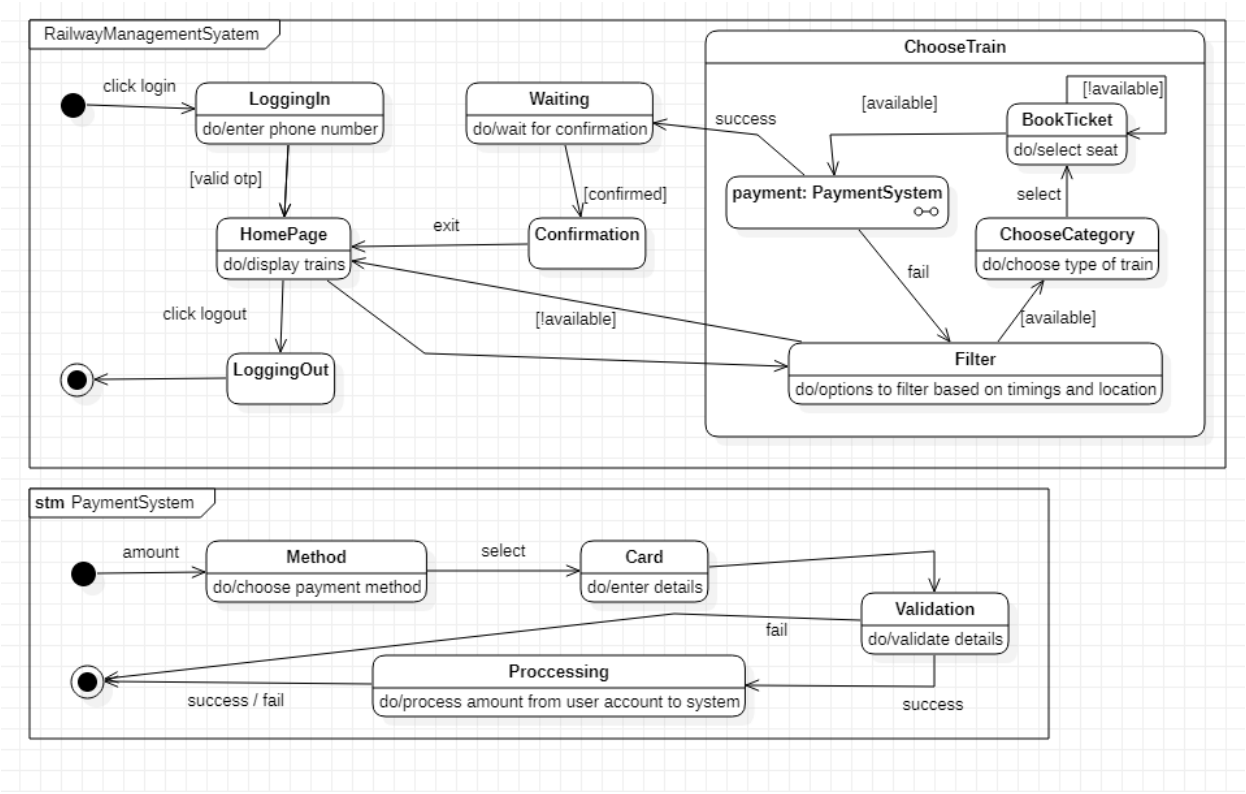
Justification: The reservation system is linked by association and coach and engine are a part of the train but can exist without it as well as it is an aggregation. Administrator is linked to every train by train no. Reservation is generalized into waiting to RAC, Confirmed, Waiting list. Passengers make reservation and book the ticket and after the payment e-ticket is obtained hence composition.


Coach
+type: String
+material: String
+noOfBookings: integer
+addCoach()
+deleteCoach()
0..1
1
Administrator
+name: String
-password: String
+manage()
+maintainRead()
+updateAddress()
+trainNo
1..*
1
Waiting
+PNRno: integer
+display()
Locomotive
+fuel: String
+shed: String
+locomotiveNo: integer
+name: String
+locopilotName: String
+assistantLocoPilotName: String
+displayDetails()
1
1
Train
+trainNo: integer
+name: String
+noOfCoach: integer
+timings: float
+searched: boolean
-guardName: String
-addCoach()
1..*
manages
0..*
Reservation
+status: Status
+trainNo: integer
+name: String
+type: String
+displayTo()
+checkAvailability()
+reserve()
1..*
Date
+Day
+Month
+Year
+getDate()
+setDate()
1..*
+customerID
Passenger
-password: String
+name: String
+email: String
+age: integer
+idProof: ID
+searchTrain()
+bookTicket()
+cancelTicket()
+payAmount()
Ticket
+ticketNo: integer
+PNR: integer
+noOfPassengers: integer
+status: Status
+dateOfTravel: Date
+source: String
+destination: String
«enumeration»
Status
OnTime
Delayed
Stopped
RAC
+PNR: integer
+display()
Confirmed
+PNR: integer
+display()
Payment
+amount: float
+type: String
+isAwaited: boolean
«enumeration»
ID
Aadhar
VoterID
Passport
DrivingLicense
PAN
InstitutionID
iTicket
+date: Date
+amount: float
+description: String
eTicket
+amount: float
+date: Date
+description: String

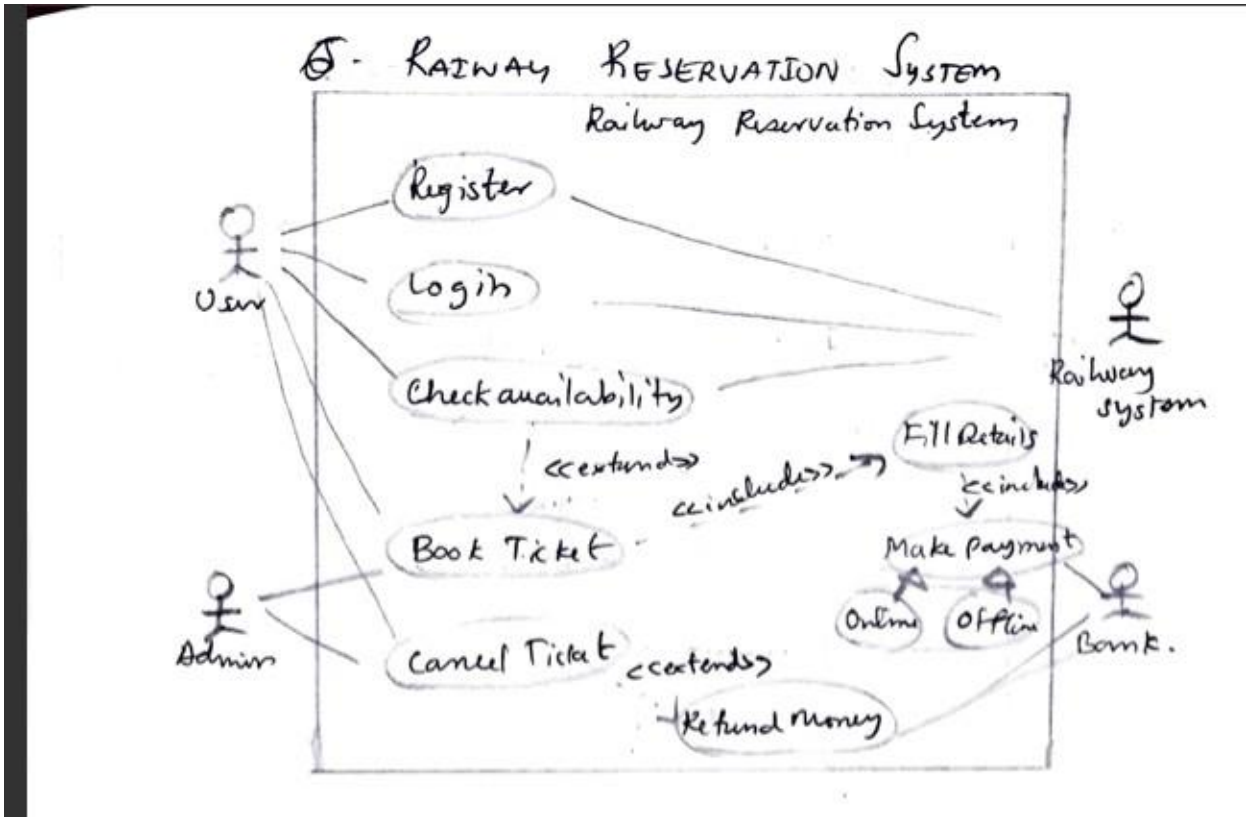
c) Advance State Diagram:



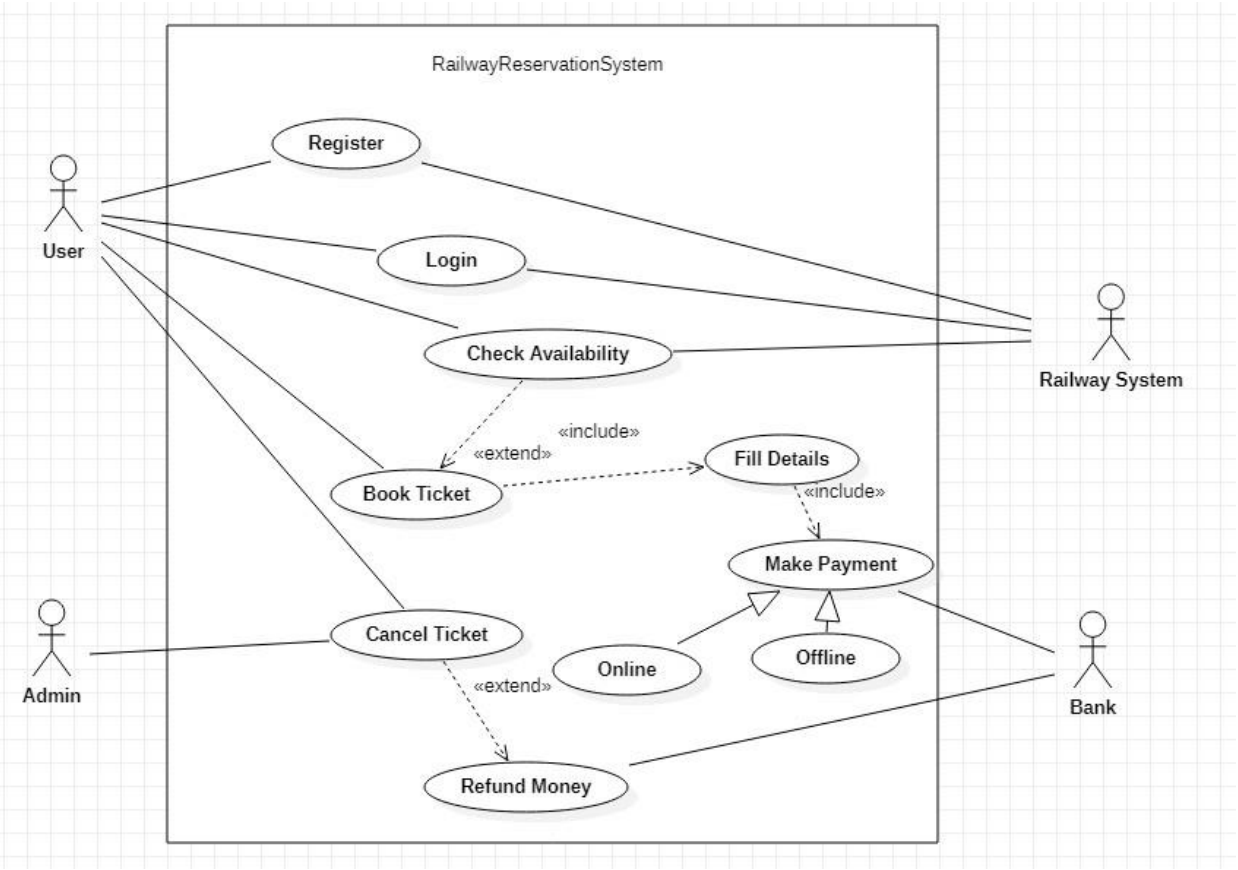
Justification: For the given case selecting train and payment for reservation is the scenario selected for the expansion of the state machine with all advance features and proper transitions mentioned for the detailed design.


RailwayManagementSyatem
click login
LoggingIn
do/enter phone number
[valid otp]
HomePage
do/display trains
click logout
LoggingOut
Waiting
do/wait for confirmation
[confirmed]
Confirmation
exit
ChooseTrain
success
[available]
[!available]
BookTicket
do/select seat
payment: PaymentSystem
select
ChooseCategory
do/choose type of train
fail
[available]
[!available]
Filter
do/options to filter based on timings and location
stm PaymentSystem
amount
Method
do/choose payment method
select
Card
do/enter details
Validation
do/validate details
fail
success
Proccessing
do/process amount from user account to system
success / fail

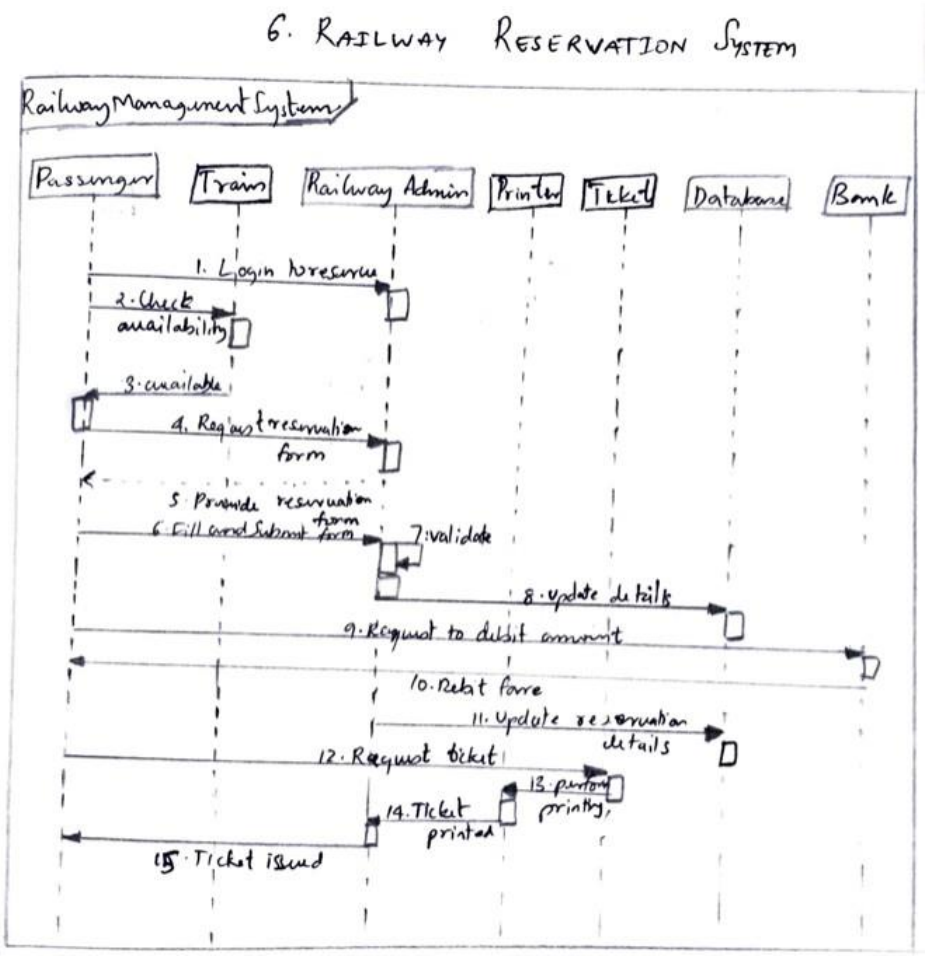
d) Advance Use Case Diagram:



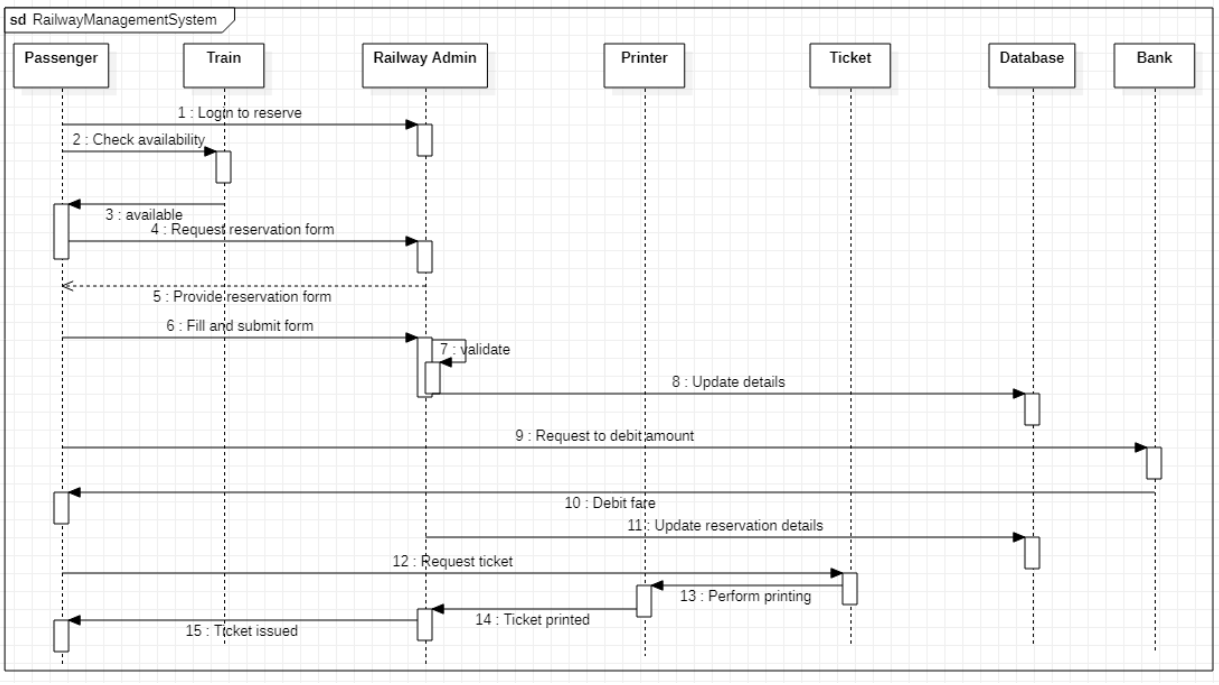
Justification: The given application all the respective functions involved for the given system are mentioned in detail and all the relation between the functions are also mentioned and also actors like user, admin, involving system and bank are mentioned and their relation is mentioned.


RailwayReservationSystem
Register
Login
Check Availability
Book Ticket
Fill Details
Make Payment
Cancel Ticket
Online
Offline
Refund Money
«include»
«extend»
«include»
«extend»
User
Admin
Railway System
Bank

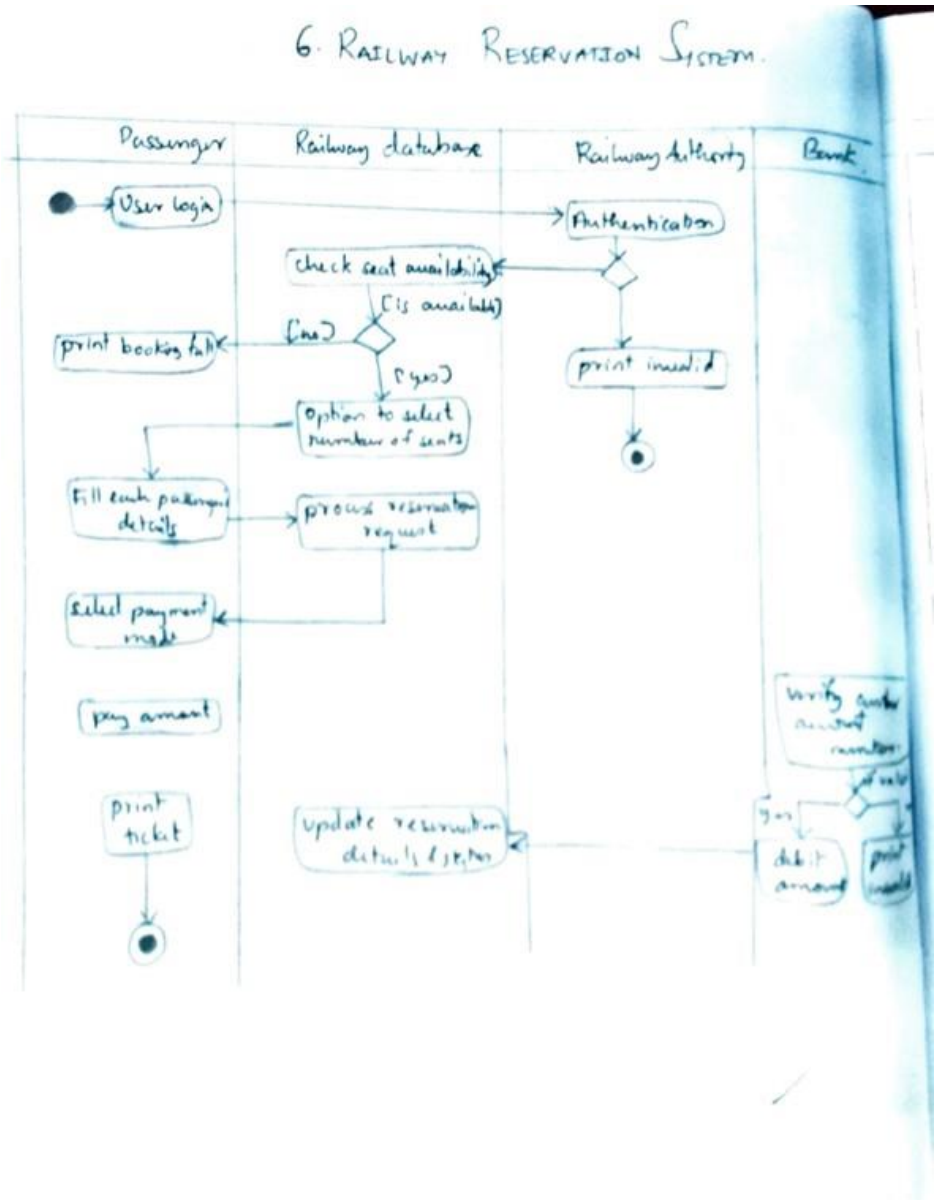
e) Sequence Diagram:



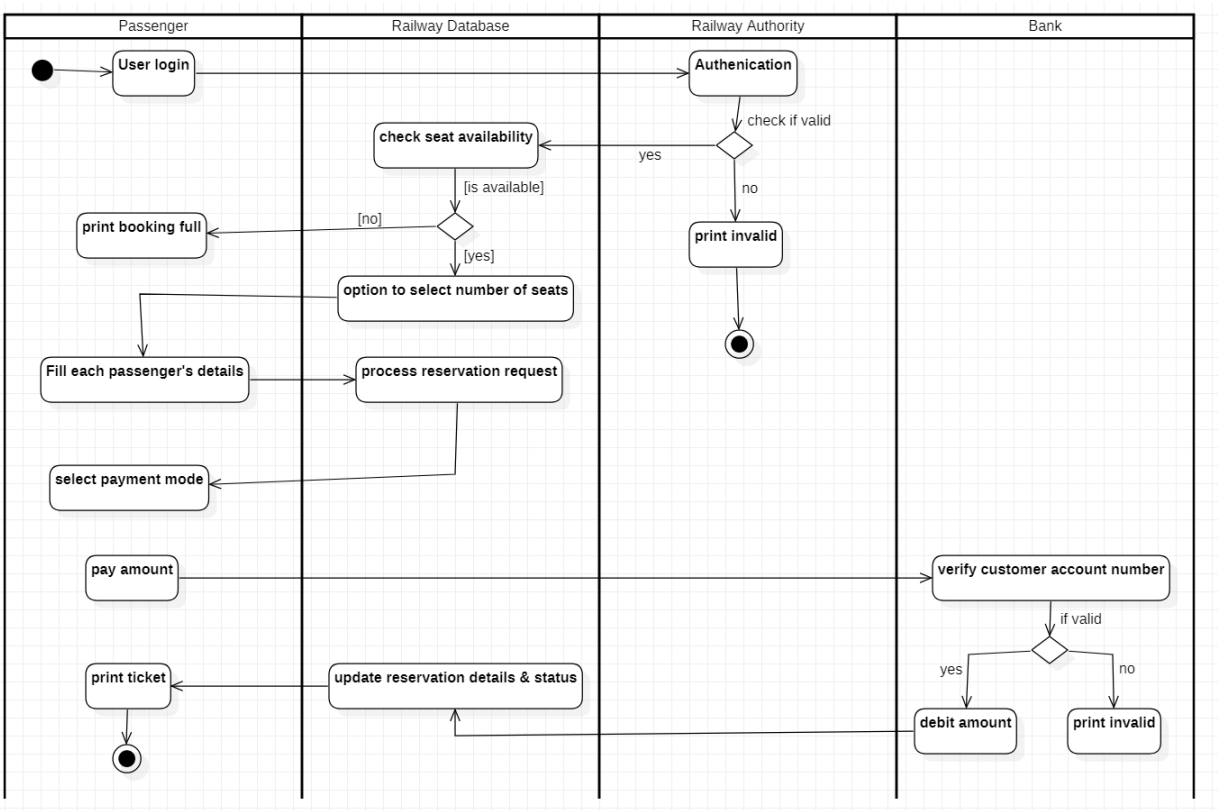
Justification: The given model the complete interacts and steps of various procedures taking place for the reservation system, from login to selecting to update details to payment and issue the ticket for the passenger.


sd RailwayManagementSystem
Passenger
Train
Railway Admin
Printer
Ticket
Database
Bank
1 : Login to reserve
2 : Check availability
3 : available
4 : Request reservation form
5 : Provide reservation form
6 : Fill and submit form
7 : Validate
8 : Update details
9 : Request to debit amount
10 : Debit fare
11 : Update reservation details
12 : Request ticket
13 : Perform printing
14 : Ticket printed
15 : Ticket issued

f) Activity Diagram:

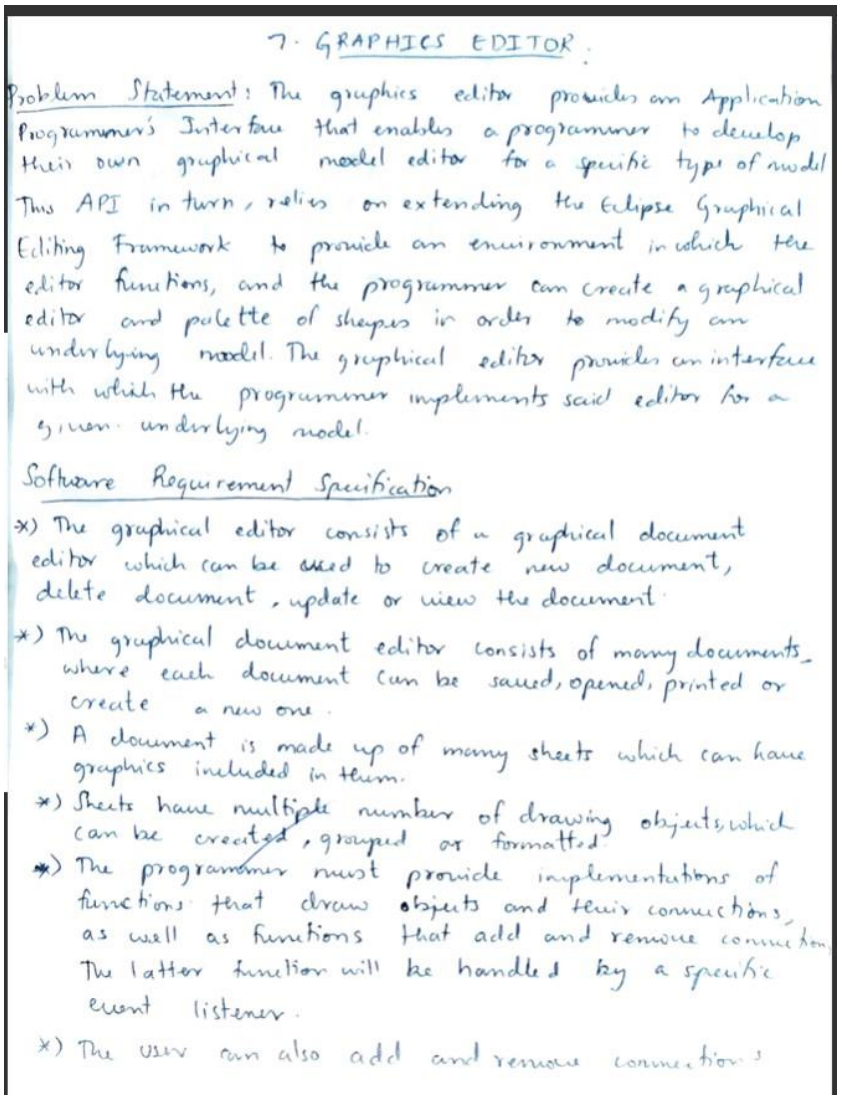


Justification: The interaction of the passengers with the system is shown in detail with respect to the railway database, authority, payment with all conditions mentioned


Passenger
Railway Database
Railway Authority
Bank
User login
Authenication
check if valid
check seat availability
yes
no
[is available]
print booking full
[no]
print invalid
[yes]
option to select number of seats
Fill each passenger's details
process reservation request
select payment mode
pay amount
verify customer account number
if valid
yes
no
print ticket
update reservation details & status
debit amount
print invalid

7. Graphics Editor-

a) SRS:



7. GRAPHICS EDITOR.

Problem Statement: The graphics editor provides an Application Programmer's Interface that enables a programmer to develop their own graphical model editor for a specific type of model. This API in turn, relies on extending the Eclipse Graphical Editing Framework to provide an environment in which the editor functions, and the programmer can create a graphical editor and palette of shapes in order to modify an underlying model. The graphical editor provides an interface with which the programmer implements said editor for a given underlying model.

Software Requirement Specification

*) The graphical editor consists of a graphical document editor which can be used to create new document, delete document, update or view the document.

*) The graphical document editor consists of many documents, where each document can be saved, opened, printed or create a new one.

*) A document is made up of many sheets which can have graphics included in them.

*) Sheets have multiple number of drawing objects which can be created, grouped or formatted.

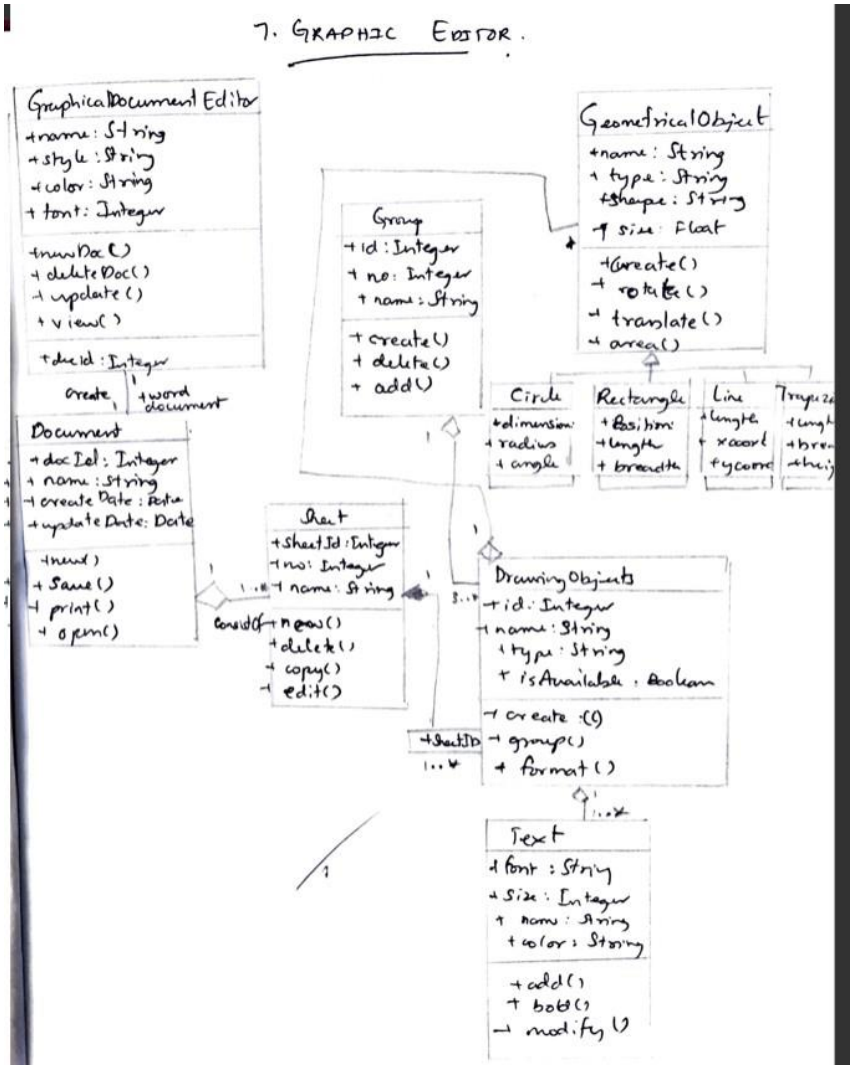
*) The programmer must provide implementations of functions that draw objects and their connections, as well as functions that add and remove connection. The latter function will be handled by a specific event listener.

*) The user can also add and remove connections between these objects as needed using the palette supplied, thus modifying the underlying model.

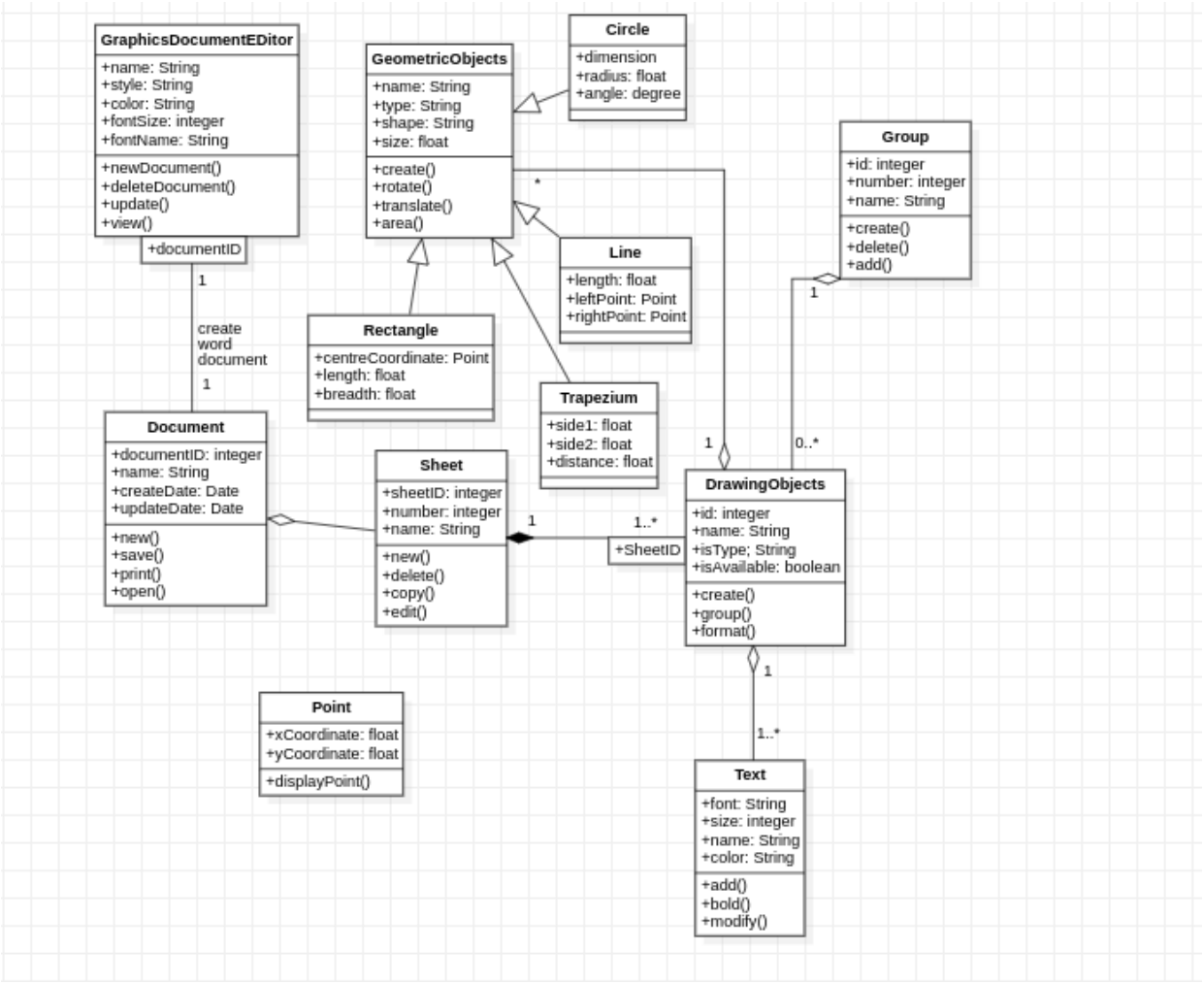
*) Each sheet contains drawing objects, including text, geometrical objects and groups. A group is simply a set of drawing objects.

*) A geometrical object includes circle, ellipse, rectangles, lines and squares, trapeziums which are identified by their respective constraints.

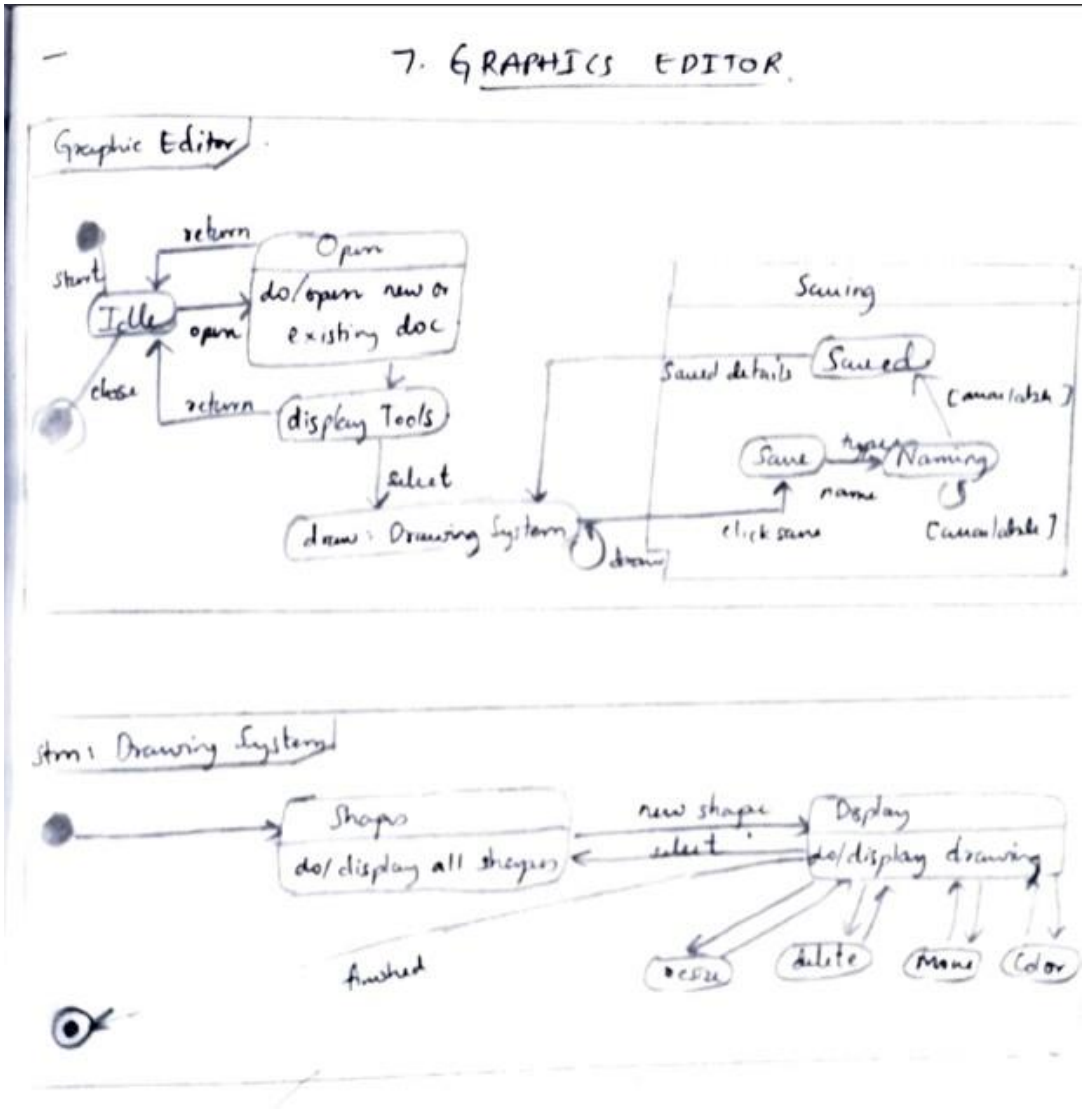
b) Advance Class Diagram:



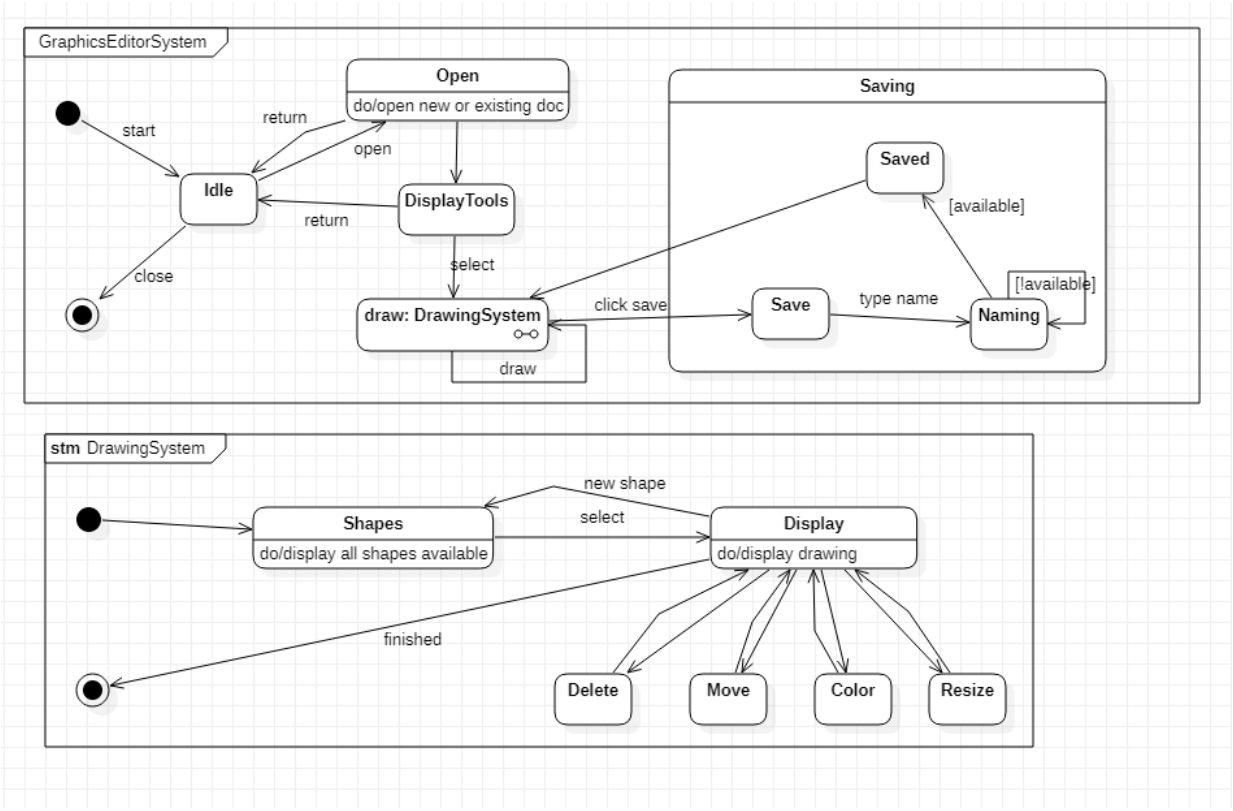
Justification: Every graphic editor has a document with which doc id for association. Document is composed of sheets which are composed of drawing objects. Which has text, geometrical objects which is generalized into several categories , groups which are linked by aggregation.


GraphicsDocumentEDitor
+name: String
+style: String
+color: String
+fontSize: integer
+fontName: String
+newDocument()
+deleteDocument()
+update()
+view()
+documentID
1
create
word
document
1
Document
+documentID: integer
+name: String
+createDate: Date
+updateDate: Date
+new()
+save()
+print()
+open()
GeometricObjects
+name: String
+type: String
+shape: String
+size: float
+create()
+rotate()
+translate()
+area()
Circle
+dimension
+radius: float
+angle: degree
Line
+length: float
+leftPoint: Point
+rightPoint: Point
Rectangle
+centreCoordinate: Point
+length: float
+breadth: float
Trapezium
+side1: float
+side2: float
+distance: float
Sheet
+sheetID: integer
+number: integer
+name: String
+new()
+delete()
+copy()
+edit()
1
1..*
+SheetID
Point
+xCoordinate: float
+yCoordinate: float
+displayPoint()
*
1
DrawingObjects
+id: integer
+name: String
+isType; String
+isAvailable: boolean
+create()
+group()
+format()
0..*
Group
+id: integer
+number: integer
+name: String
+create()
+delete()
+add()
1
1
1..*
Text
+font: String
+size: integer
+name: String
+color: String
+add()
+bold()
+modify()

c) Advance State Diagram:

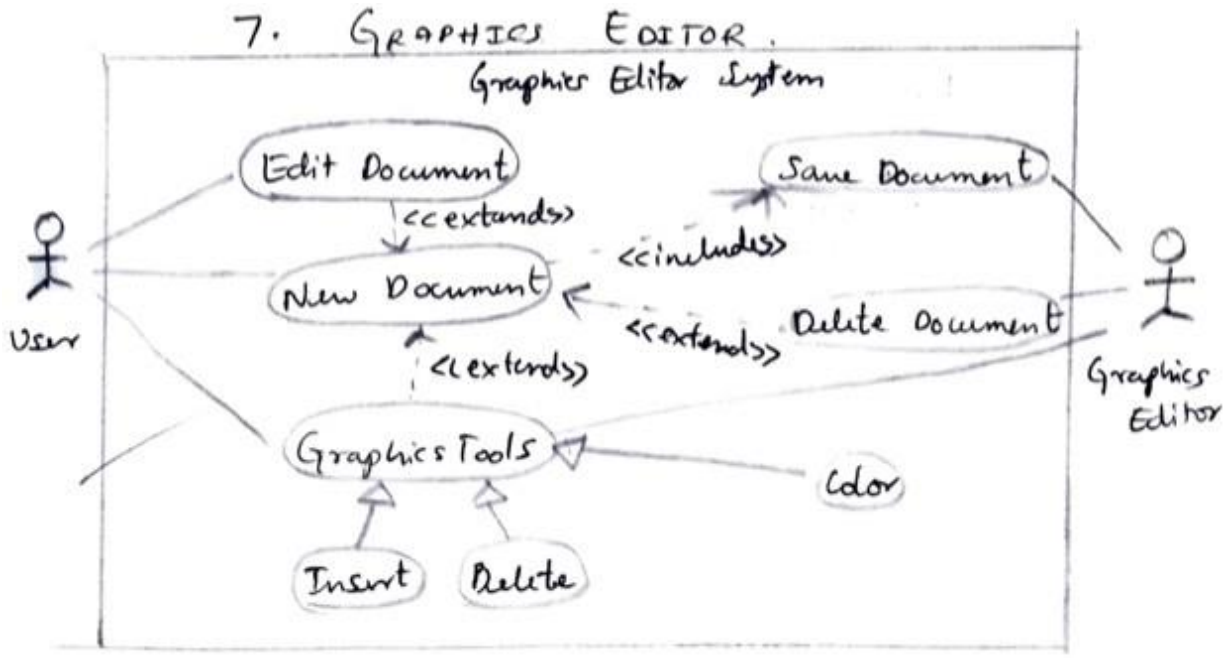


Justification: For the given case, the state machine is drawn with saving a file and drawing system as the scenario with detailed description with advance features included, for proper understanding of system with appropriate transitions and conditions.

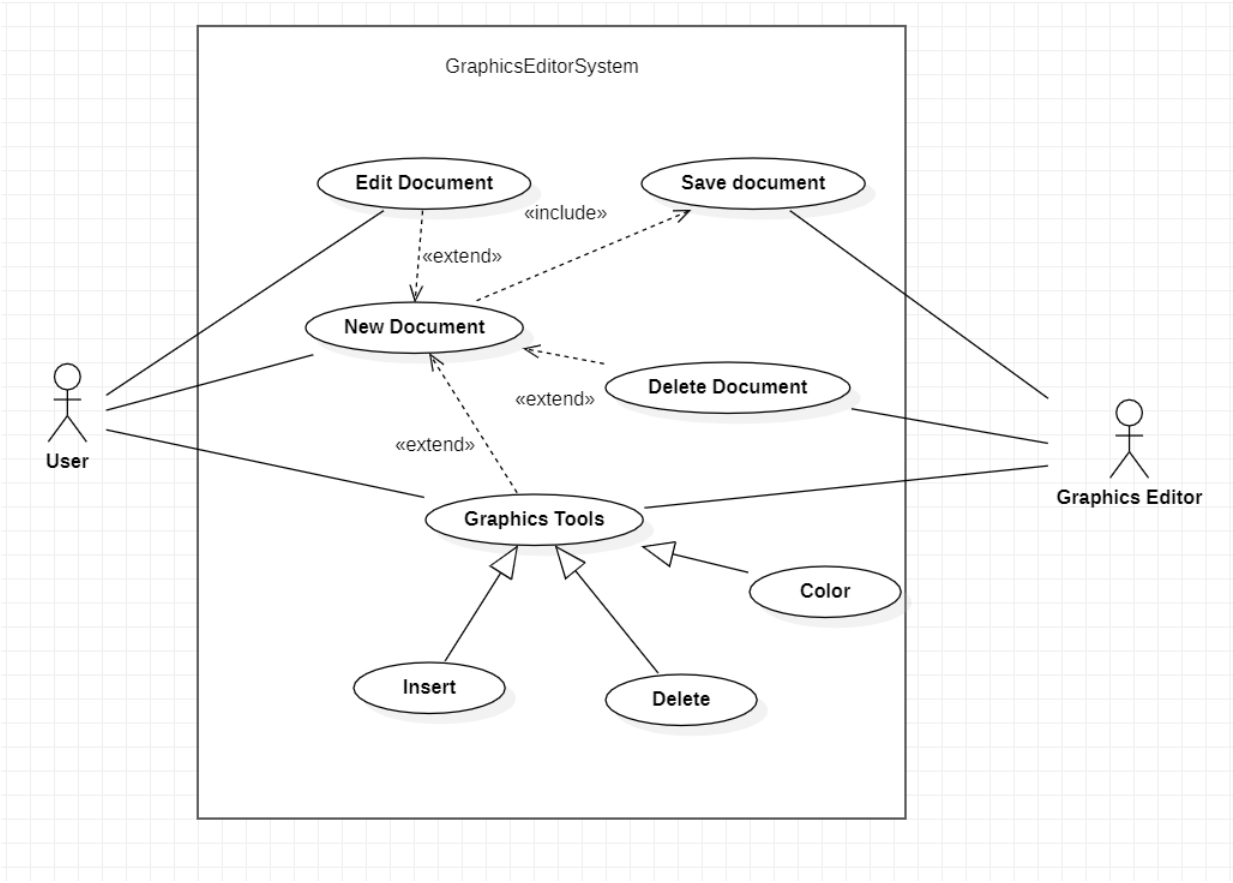

GraphicsEditorSystem
Open
do/open new or existing doc
start
return
open
Idle
DisplayTools
return
close
select
draw: DrawingSystem
draw
Saving
Saved
[available]
[!available]
click save
Save
type name
Naming

stm DrawingSystem
new shape
Shapes
do/display all shapes available
select
Display
do/display drawing
finished
Delete
Move
Color
Resize

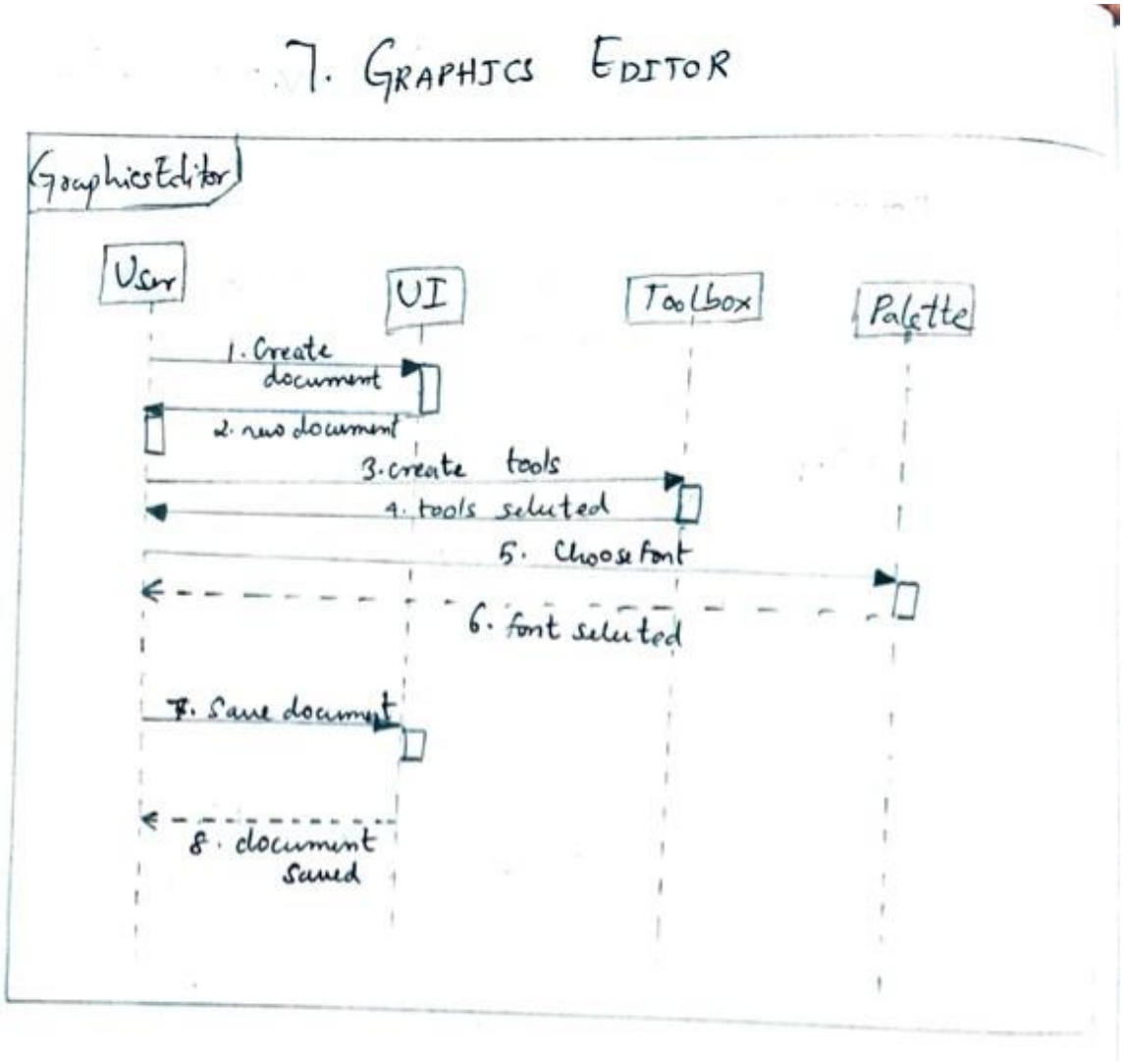
d) Advance Use Case Diagram:



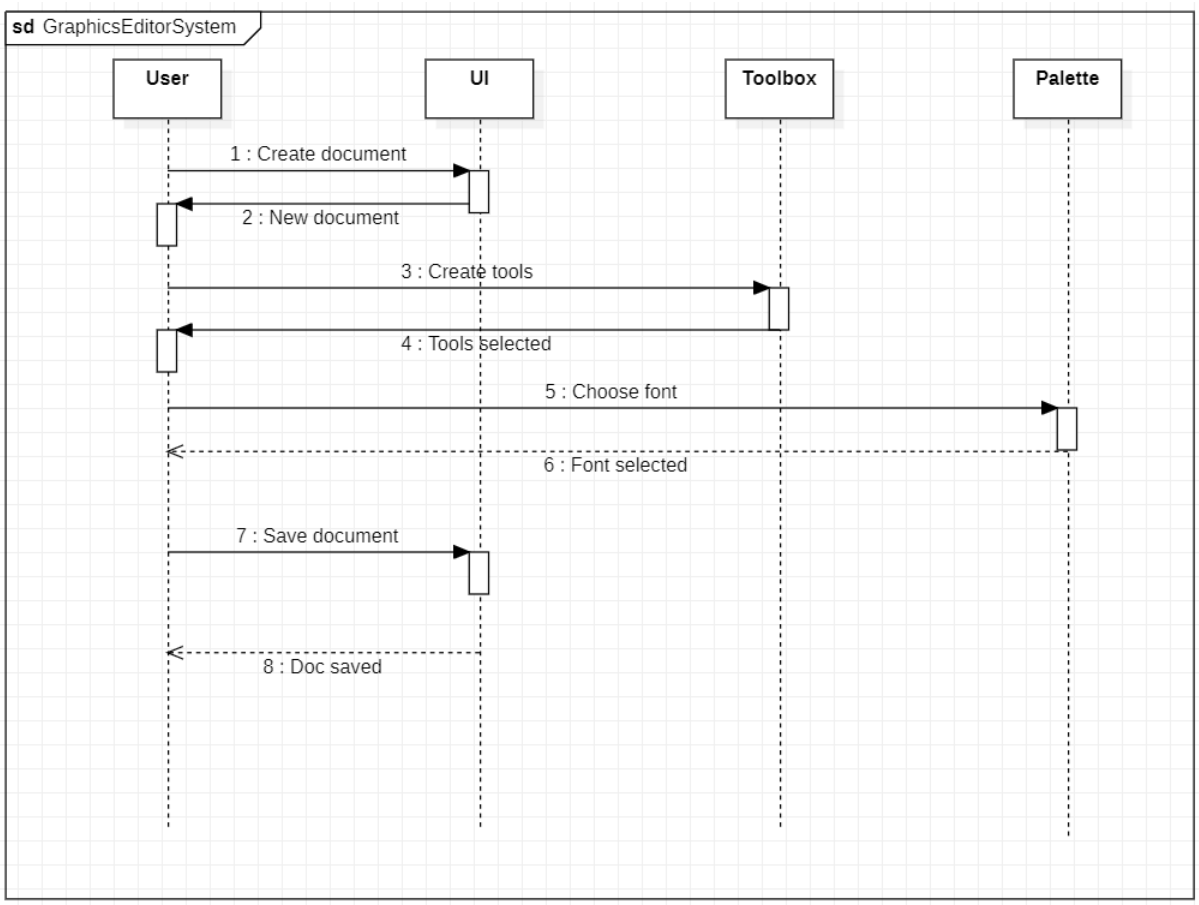
Justification: All the functions for the given application are mentioned and all the relations with the functions are given with all the respective actors like user, graphics system. They are given with appropriate functionalities.


GraphicsEditorSystem
Edit Document
Save document
«include»
«extend»
New Document
Delete Document
«extend»
«extend»
User
Graphics Editor
Graphics Tools
Color
Insert
Delete

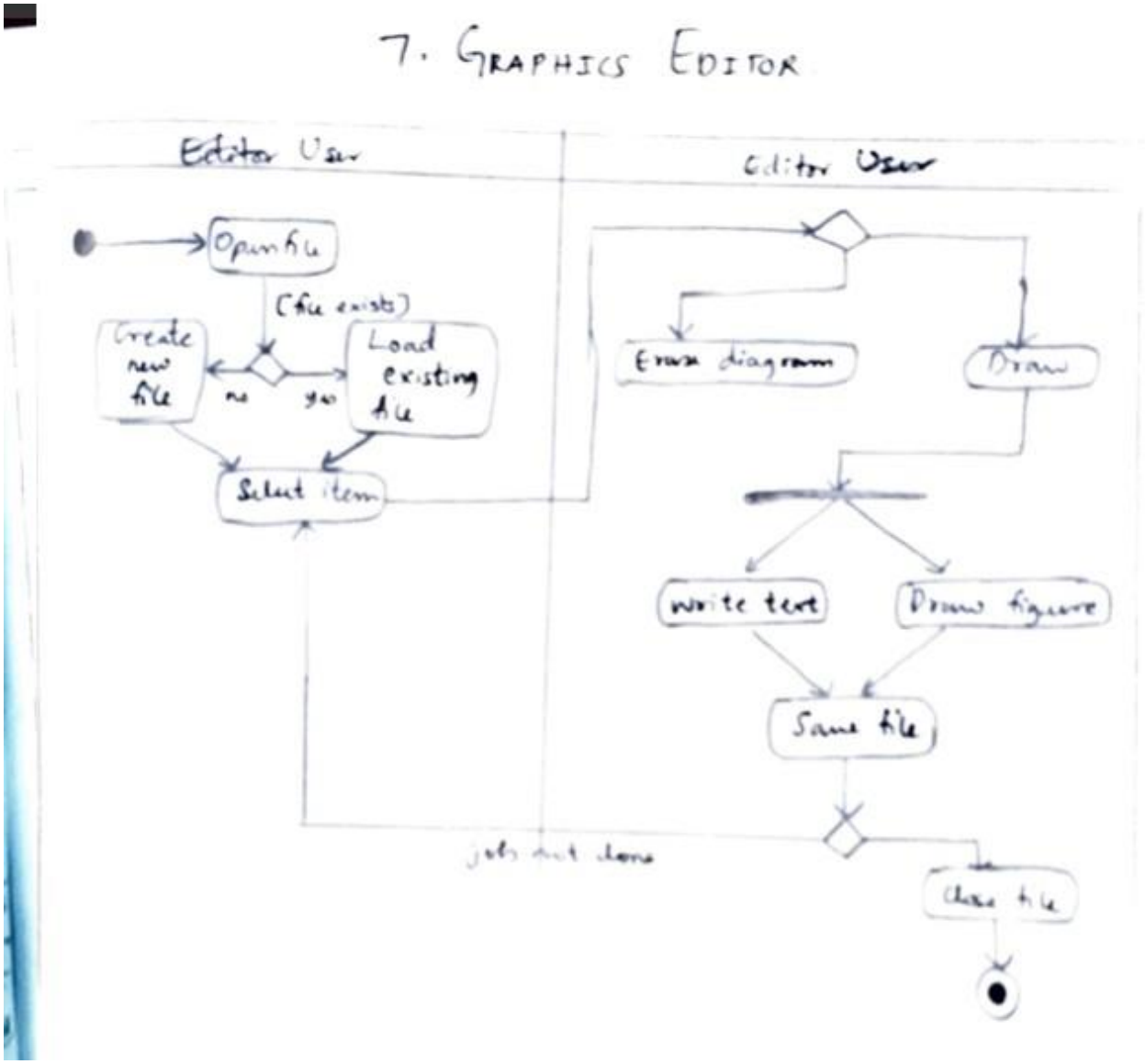
e) Sequence Diagram:



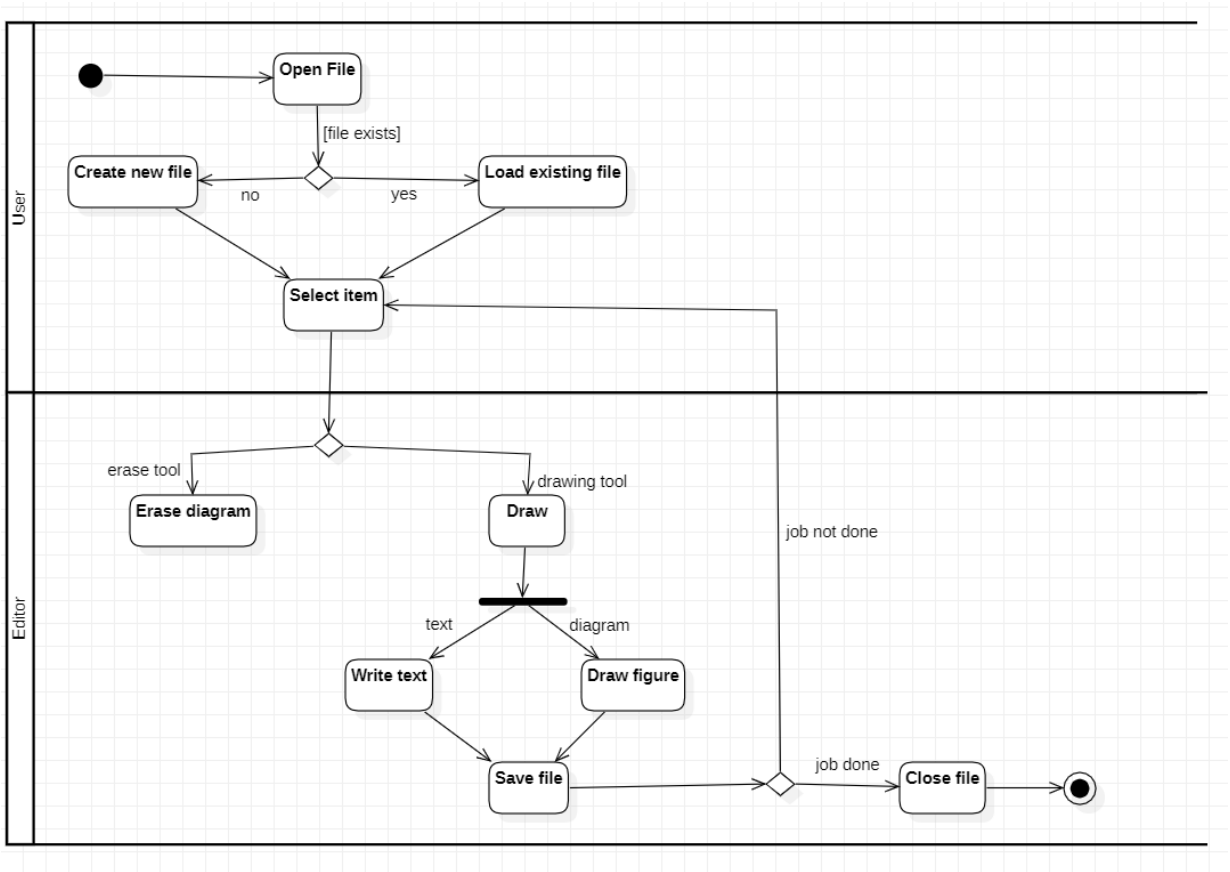
Justification: The given system gives the computer interactions of the system between users and the editor, it gives a detailed step by step procedure from creating the document to saving the document, including typing of text and addition of geometrical objects.


sd GraphicsEditorSystem
User
UI
Toolbox
Palette
1 : Create document
2 : New document
3 : Create tools
4 : Tools selected
5 : Choose font
6 : Font selected
7 : Save document
8 : Doc saved

f) Activity Diagram:



Justification: The given activity model explains the complete interaction between of the user with the editor and how the various functions are performed, from the diagrams to saving the file.


User
Editor
Open File
[file exists]
Create new file
no
yes
Load existing file
Select item
erase tool
drawing tool
Erase diagram
Draw
job not done
text
diagram
Write text
Draw figure
Save file
job done
Close file