VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

OBJECT ORIENTED MODELING AND DESIGN

Submitted by

DISHA N (1BM19CS051)

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 MODELING AND DESIGN** " carried out by **DISHA N (1BM19CS051),** who is 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 MODELING AND DESIGN - (20CS6PCOMD)** work prescribed for the said degree.

Sheetal V A
Assistant Professor
Department of CSE
BMSCE, Bengaluru

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

`

Index Sheet

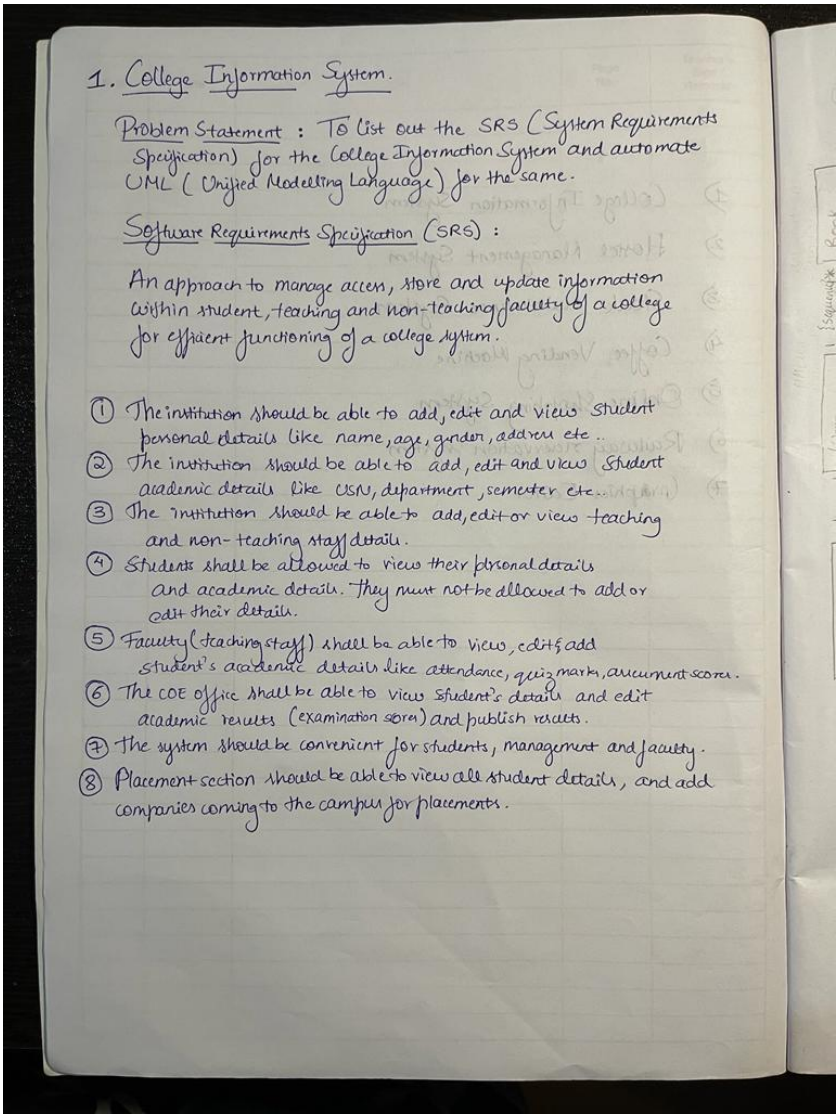
Sl. No.	Experiment Title	Page No.
1.	College information system	4-9
2.	Hostel management system	10-15
3.	Stock management system	16-21
4.	Coffee vending machine	22-27
5.	Online shopping system	28-33
6.	Railway reservation system	34-39
7.	Graphics editor system	40-45

Course Outcome

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

Exercise 1: College Information System

1. SRS



1. College Information System.

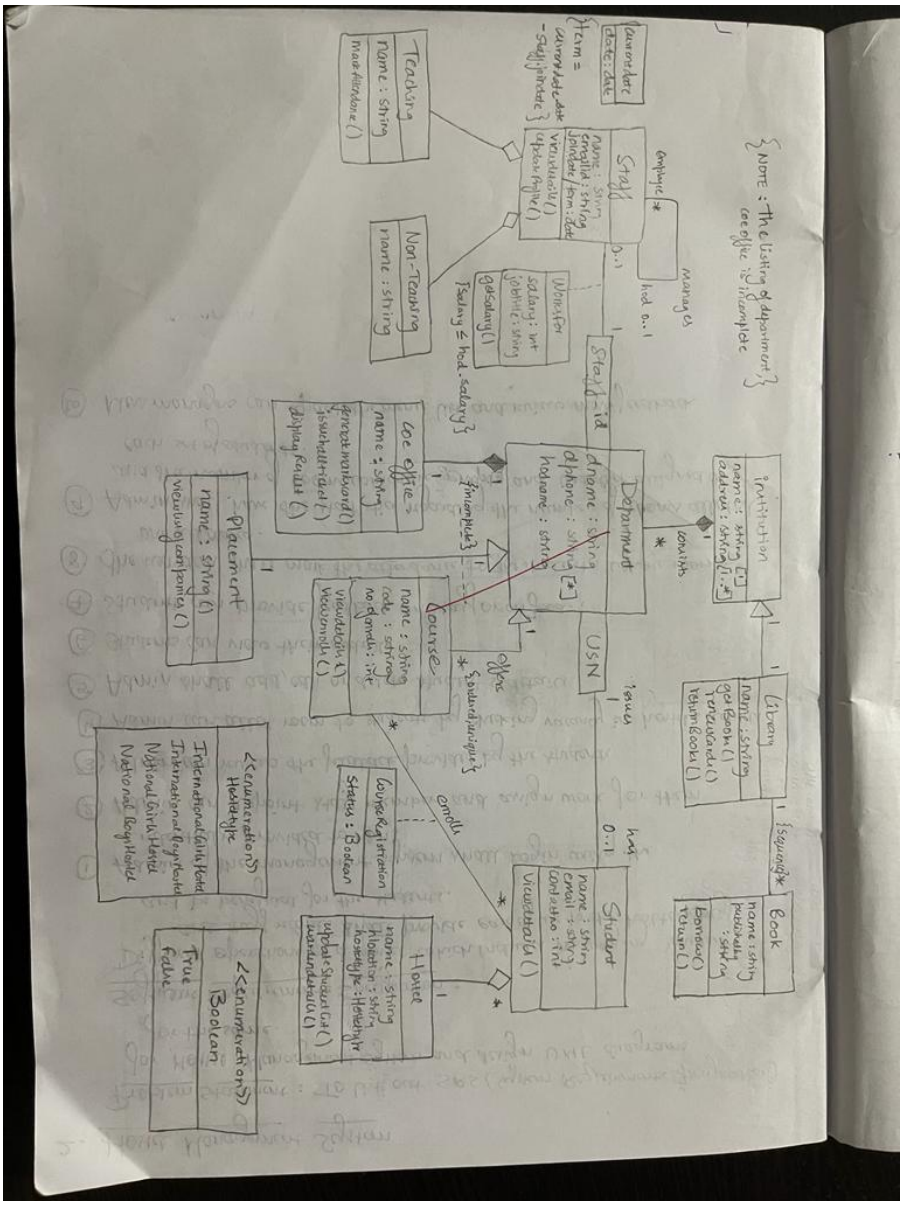
Problem Statement : To list out the SRS (System Requirements Specification) for the College Information System and automate UML (Unified Modelling Language) for the same.

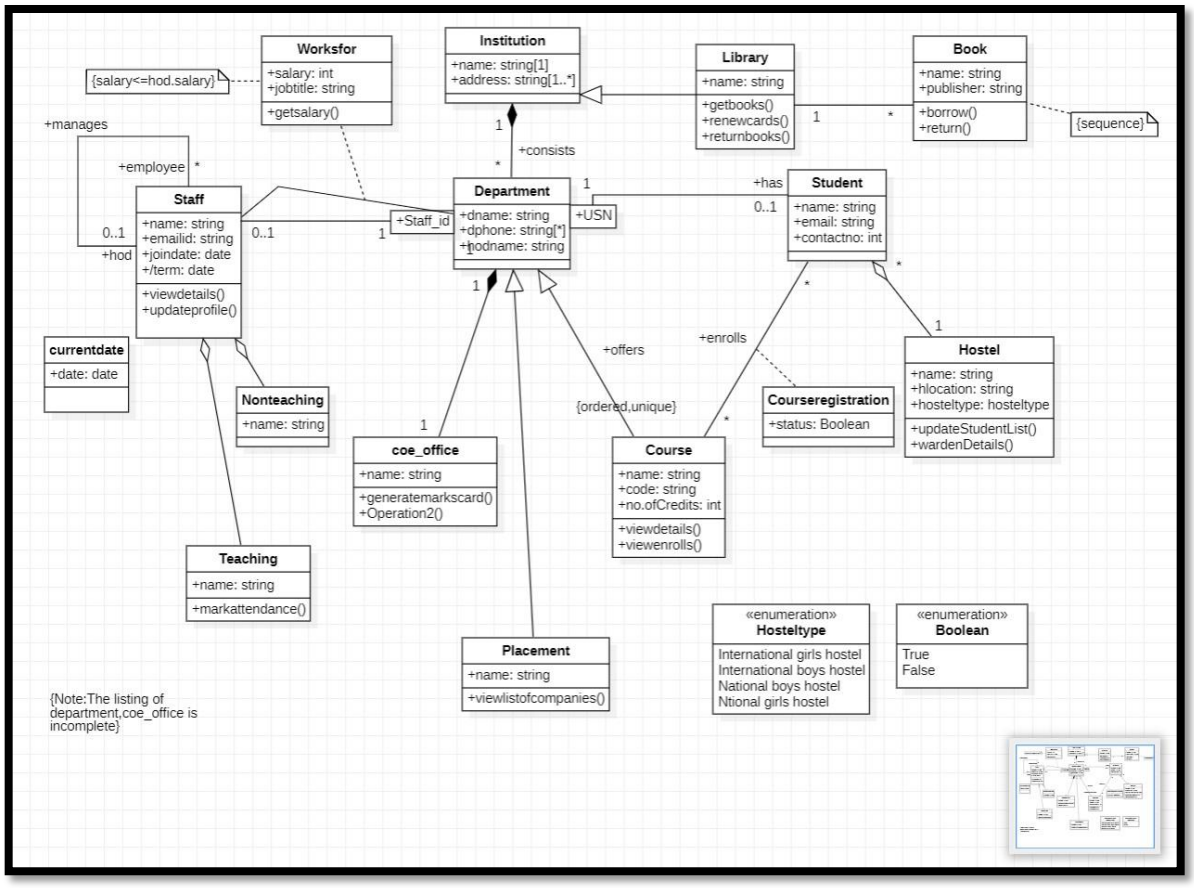
Software Requirements Specification (SRS) :

An approach to manage access, store and update information within student, teaching and non-teaching faculty of a college for efficient functioning of a college system.

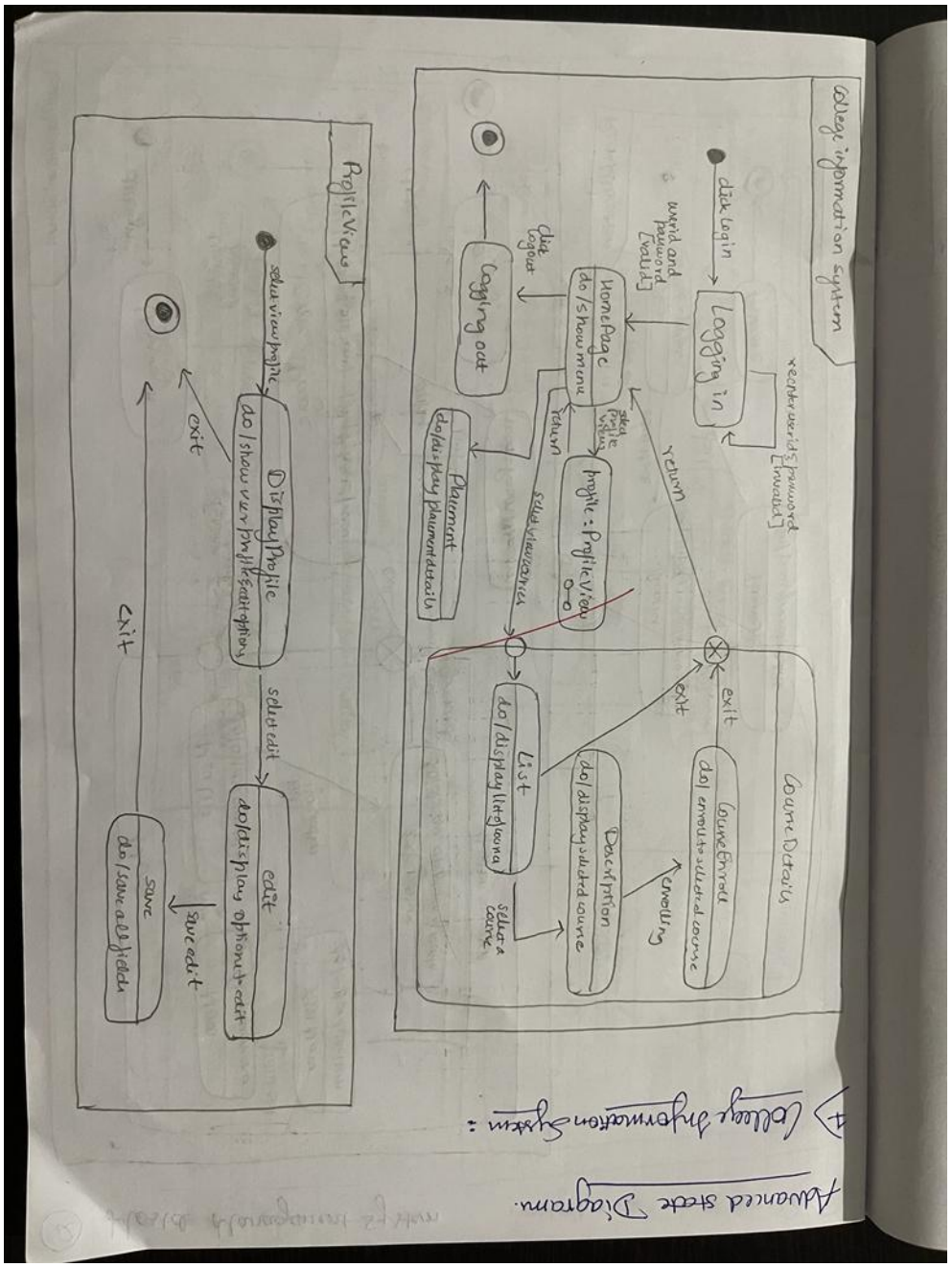
1. The institution should be able to add, edit and view student personal details like name, age, gender, address etc..
2. The institution should be able to add, edit and view student academic details like USN, department, semester etc..
3. The institution should be able to add, edit or view teaching and non-teaching staff details.
4. Students shall be allowed to view their personal details and academic details. They must not be allowed to add or edit their details.
5. Faculty (teaching staff) shall be able to view, edit & add student's academic details like attendance, quiz marks, assignment scores.
6. The COE office shall be able to view student's details and edit academic results (examination scores) and publish results.
7. The system should be convenient for students, management and faculty.
8. Placement section should be able to view all student details, and add companies coming to the campus for placements.

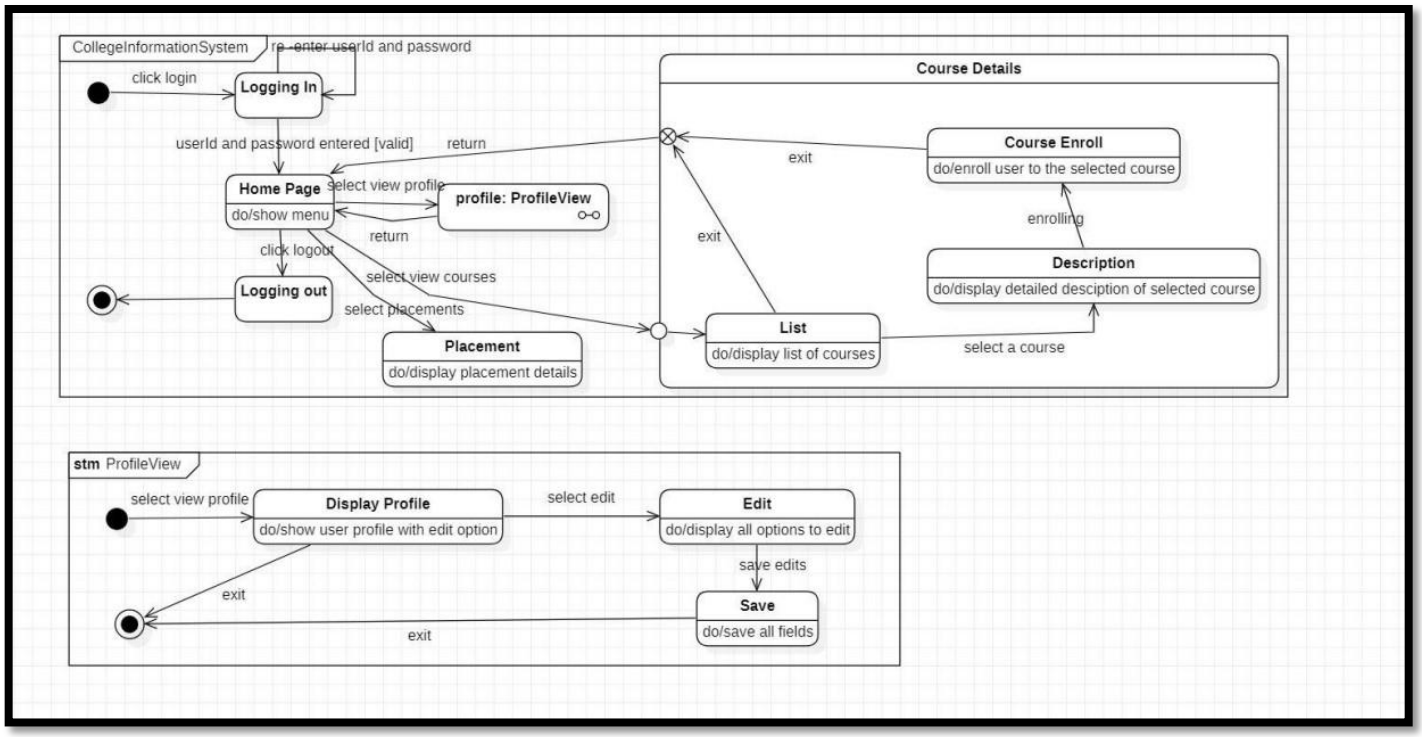
2. Advanced class diagram



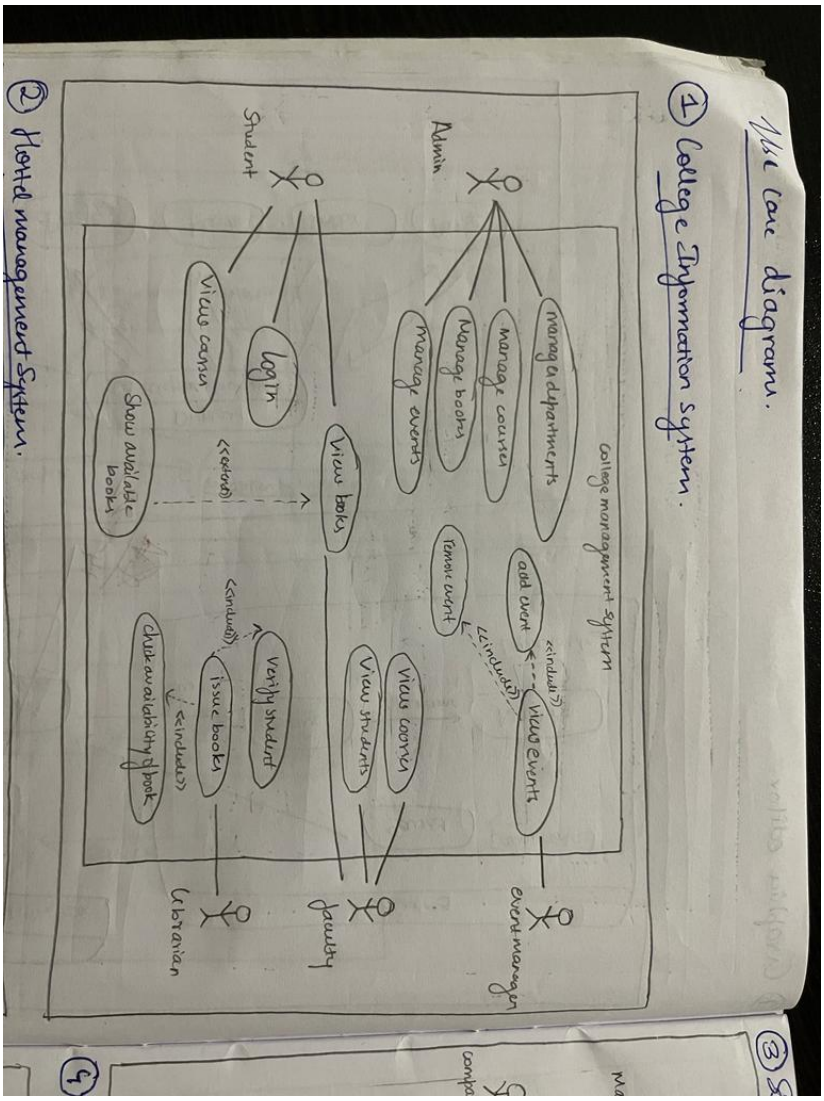

Worksfor
+salary: int
+jobtitle: string
+getsalary()
{salary<=hod.salary}
Institution
+name: string[1]
+address: string[1..*]
Library
+name: string
+getbooks()
+renewcards()
+returnbooks()
Book
+name: string
+publisher: string
+borrow()
+return()
{sequence}
+manages
+employee
+hod
Staff
+name: string
+emailid: string
+joindate: date
+/term: date
+viewdetails()
+updateprofile()
+consists
Department
+dname: string
+dphone: string[*]
+hodname: string
+Staff_id
+USN
+has
Student
+name: string
+email: string
+contactno: int
currentdate
+date: date
Nonteaching
+name: string
Teaching
+name: string
+markattendance()
coe_office
+name: string
+generatemarkscard()
+Operation2()
+offers
{ordered,unique}
Course
+name: string
+code: string
+no.ofCredits: int
+viewdetails()
+viewenrolls()
+enrolls
Courseregistration
+status: Boolean
Hostel
+name: string
+hlocation: string
+hosteltype: hosteltype
+updateStudentList()
+wardenDetails()
Placement
+name: string
+viewlistofcompanies()
«enumeration»
Hosteltype
International girls hostel
International boys hostel
National boys hostel
Ntional girls hostel
«enumeration»
Boolean
True
False
{Note:The listing of department,coe_office is incomplete}

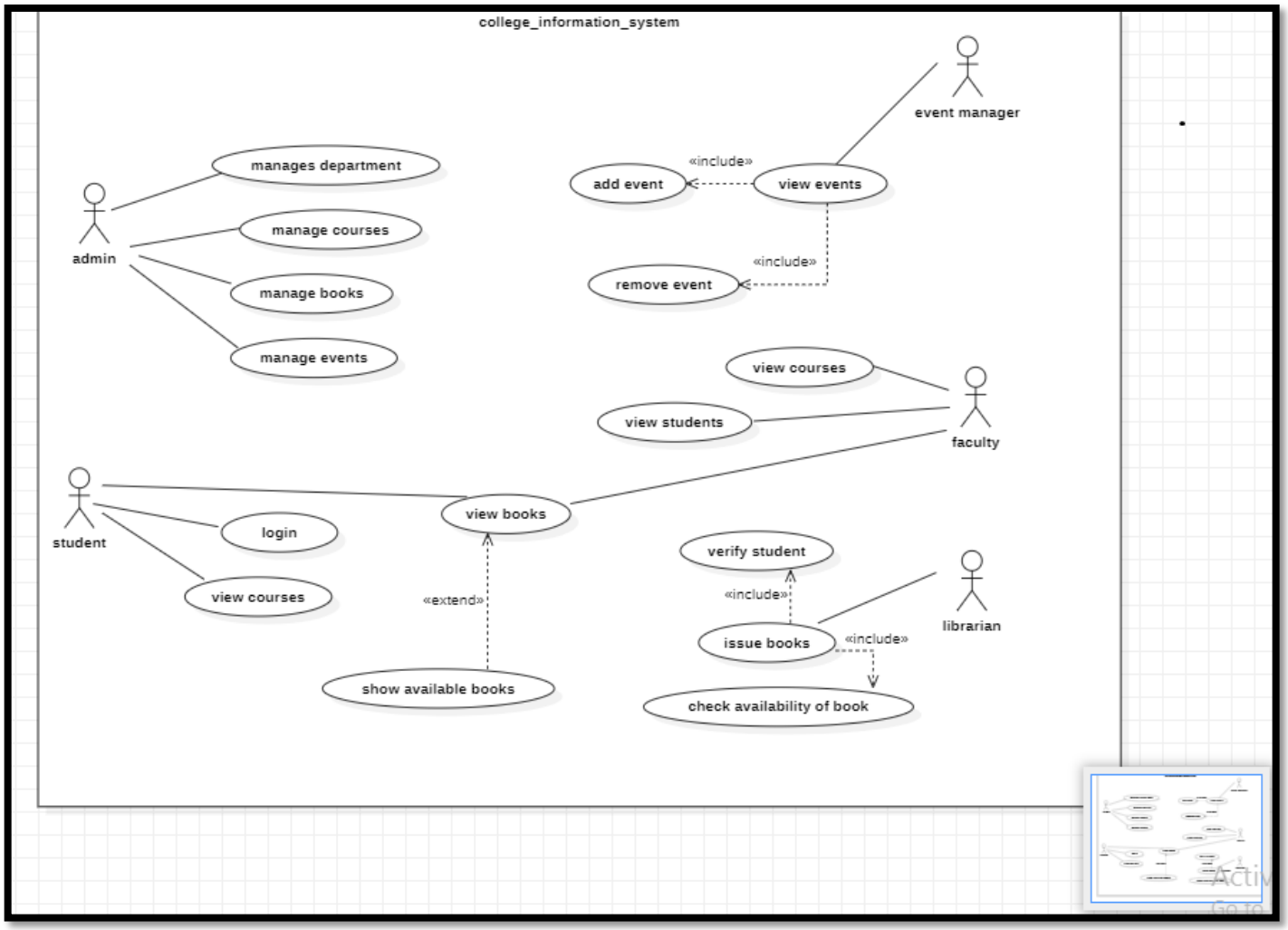
3. Advanced state diagram



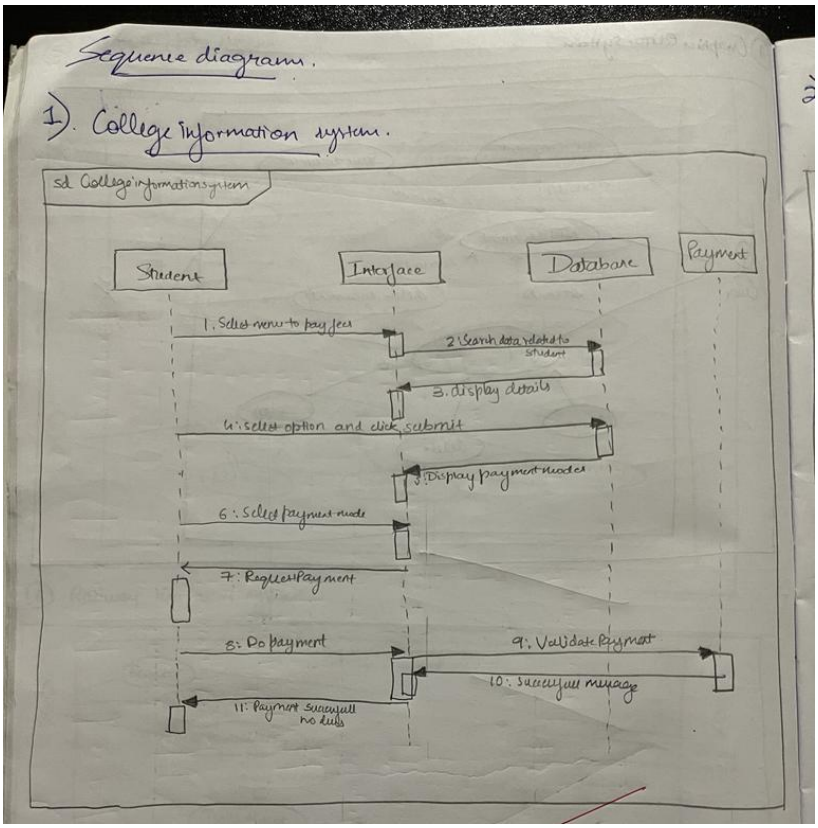

CollegeInformationSystem
re -enter userId and password
click login
Logging In
userId and password entered [valid]
return
Home Page
do/show menu
select view profile
profile: ProfileView
return
click logout
Logging out
select view courses
select placements
Placement
do/display placement details
Course 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

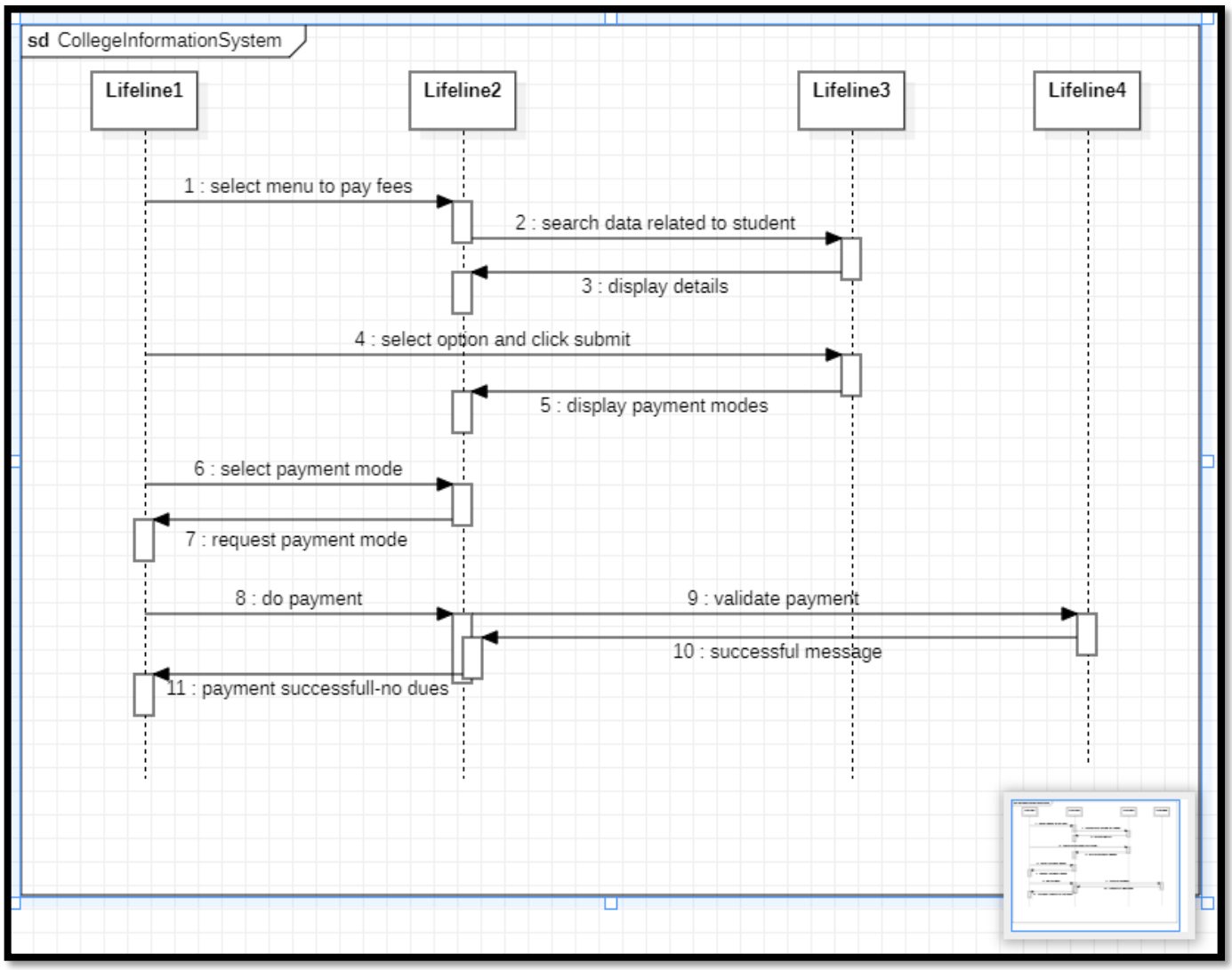
4. Advanced use case diagram



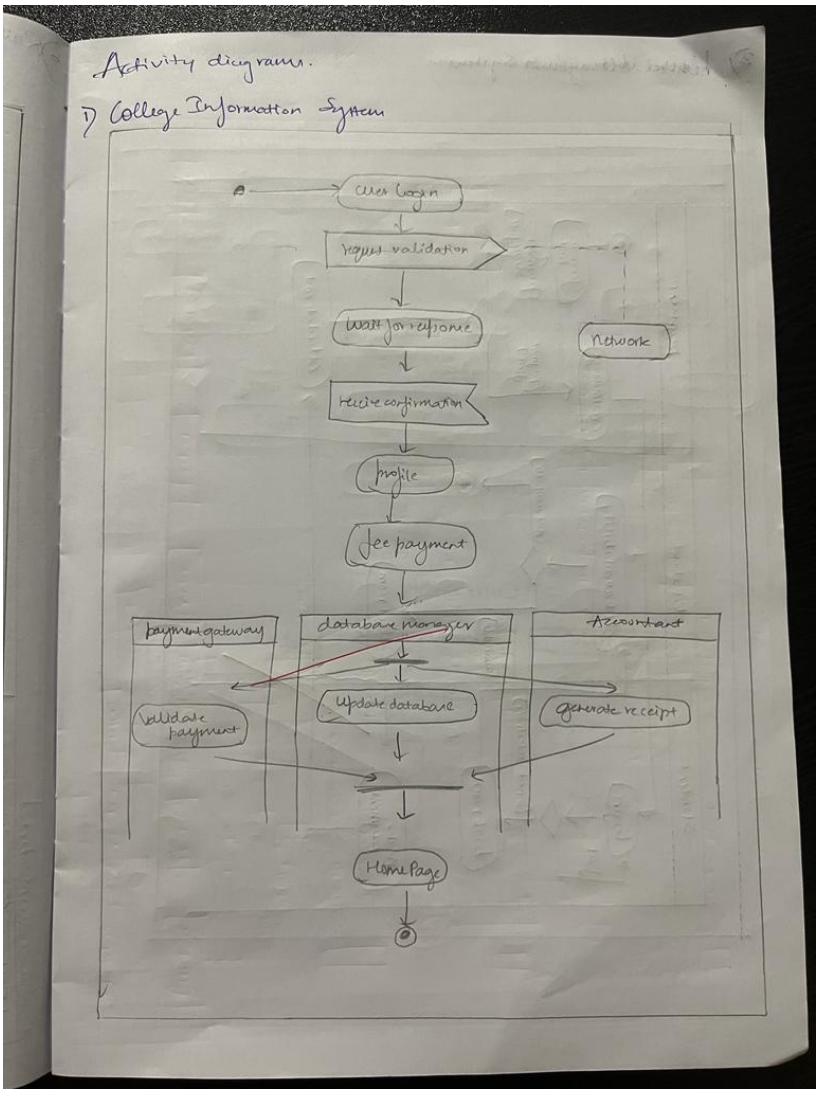

college_information_system
event manager
manages department
admin
manage courses
manage books
manage events
«include»
add event
view events
«include»
remove event
view courses
view students
faculty
student
login
view courses
view books
«extend»
show available books
verify student
«include»
issue books
«include»
check availability of book
librarian

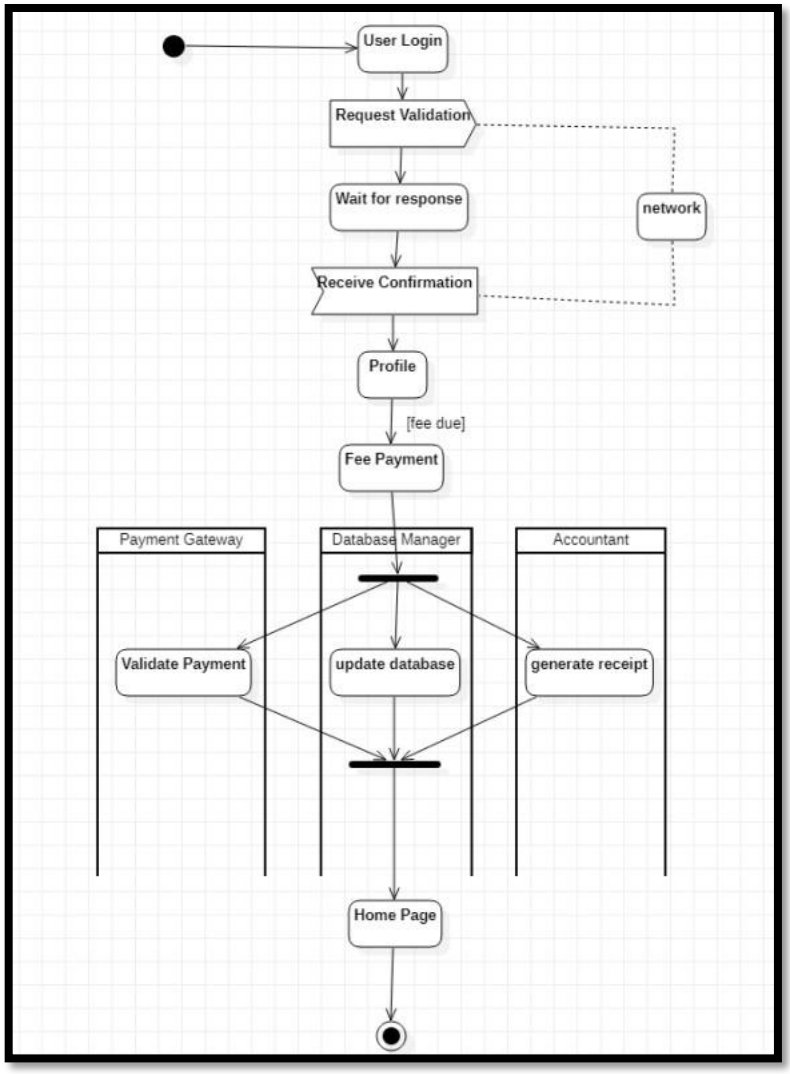
5. Advanced sequence diagram




sd CollegeInformationSystem
Lifeline1
Lifeline2
Lifeline3
Lifeline4
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 mode
8 : do payment
9 : validate payment
10 : successful message
11 : payment successfull-no dues

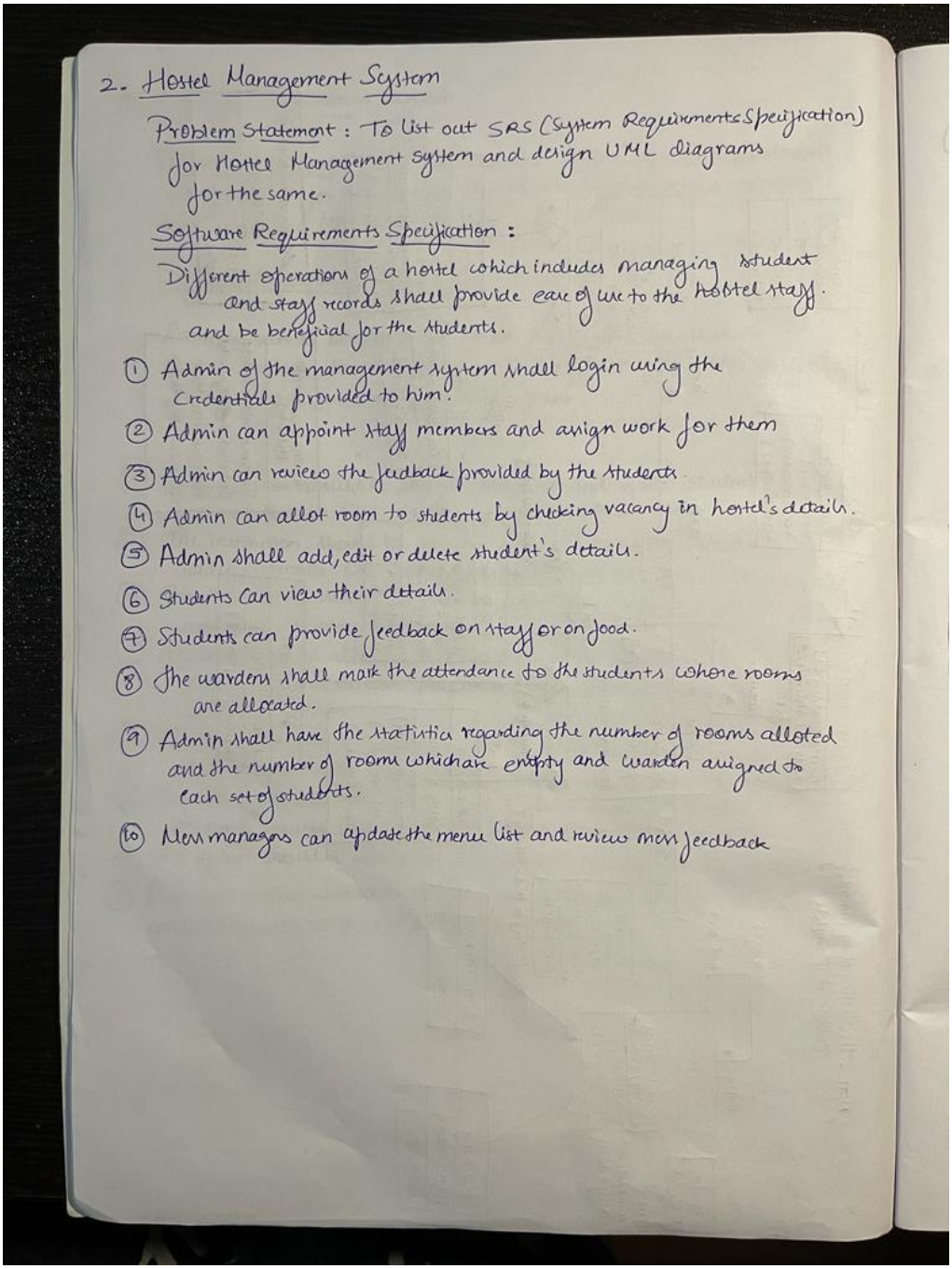
6. Advanced activity diagram




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

Exercise 2: Hostel Management System

1. SRS



2. Hostel Management System

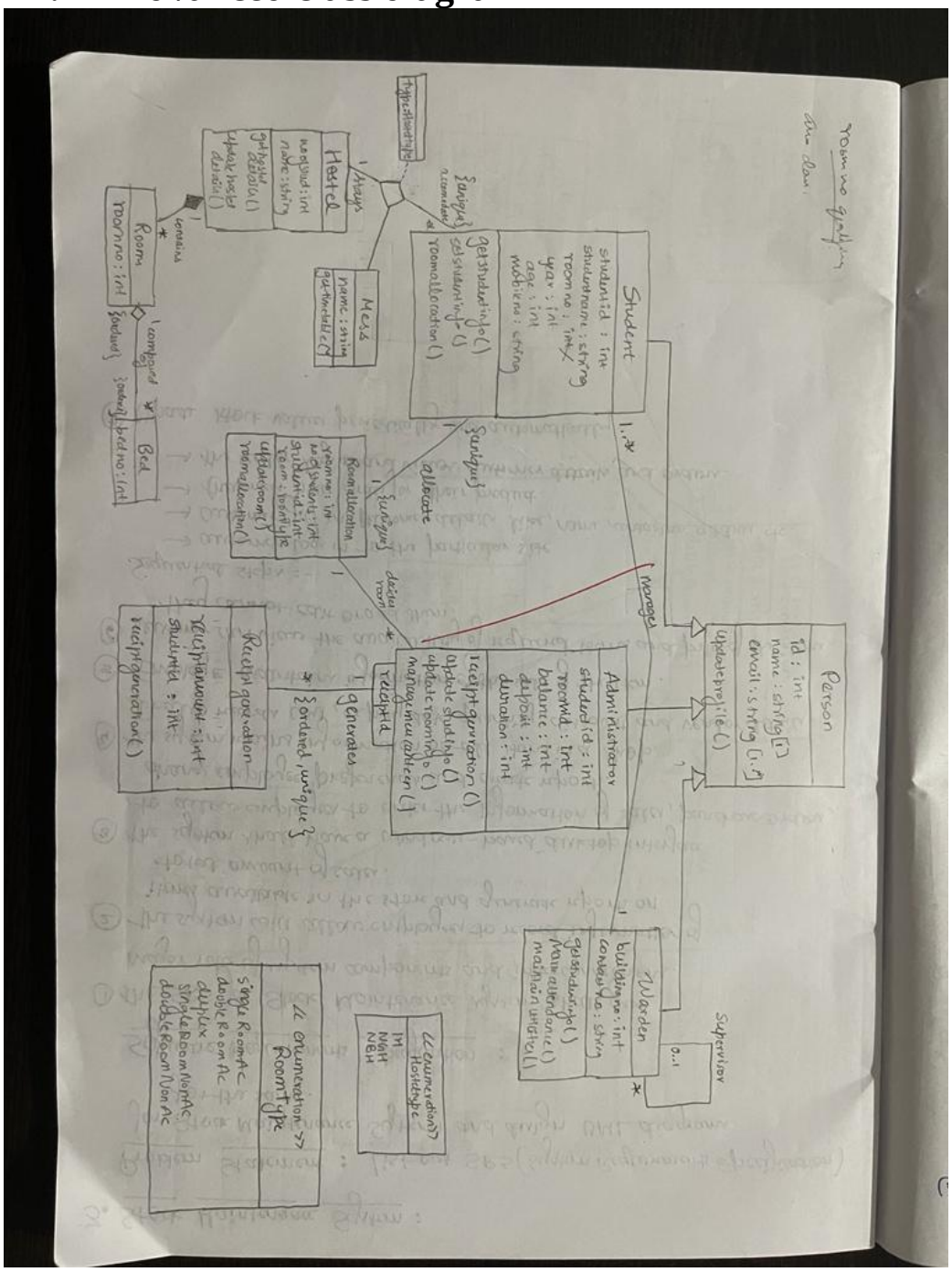
Problem Statement : To list out SRS (System Requirements Specification) for Hostel Management system and design UML diagrams for the same.

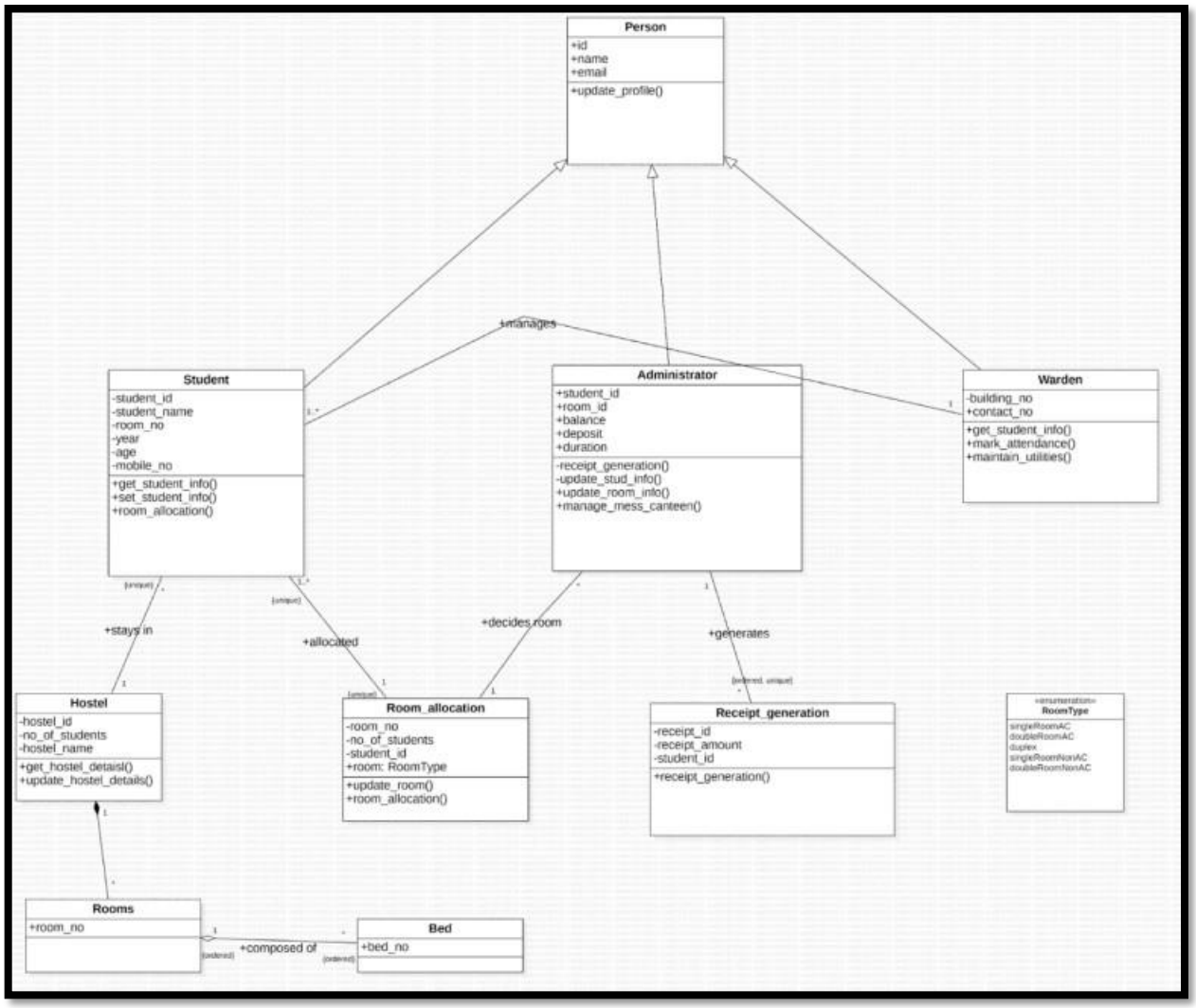
Software Requirements Specification :

Different operations of a hostel which includes managing student and staff records shall provide ease of use to the hostel staff. and be beneficial for the students.

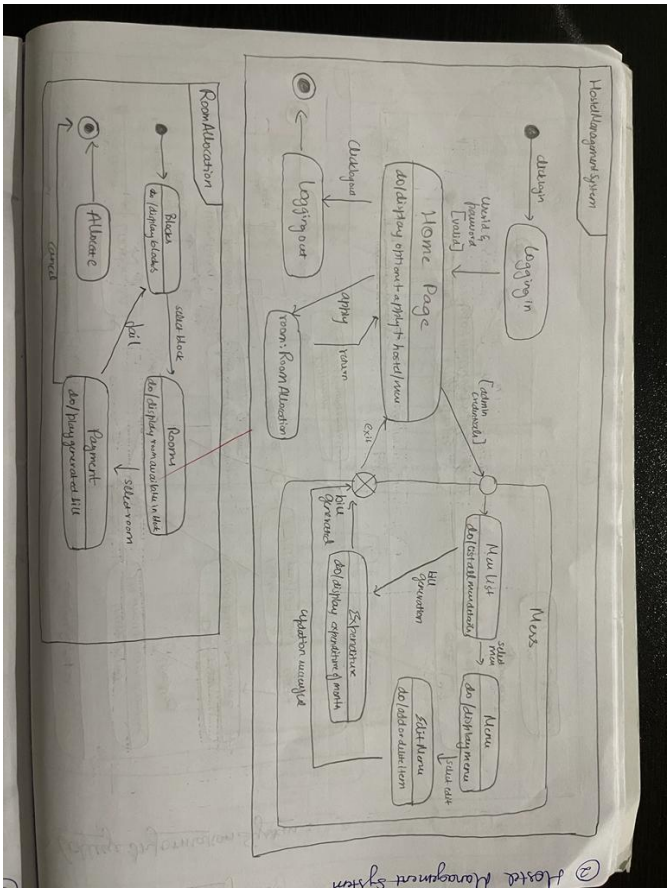
1. Admin of the management system shall login using the credentials provided to him.
2. Admin can appoint staff members and assign work for them
3. Admin can reviews the feedback provided by the students
4. Admin can allot room to students by checking vacancy in hostel's details.
5. Admin shall add, edit or delete student's details.
6. Students can view their details.
7. Students can provide feedback on staff or on food.
8. The wardens shall mark the attendance of the students whose rooms are allocated.
9. Admin shall have the statistics regarding the number of rooms alloted and the number of rooms which are empty and warden assigned to each set of students.
10. Mess managers can update the menu list and review mess feedback

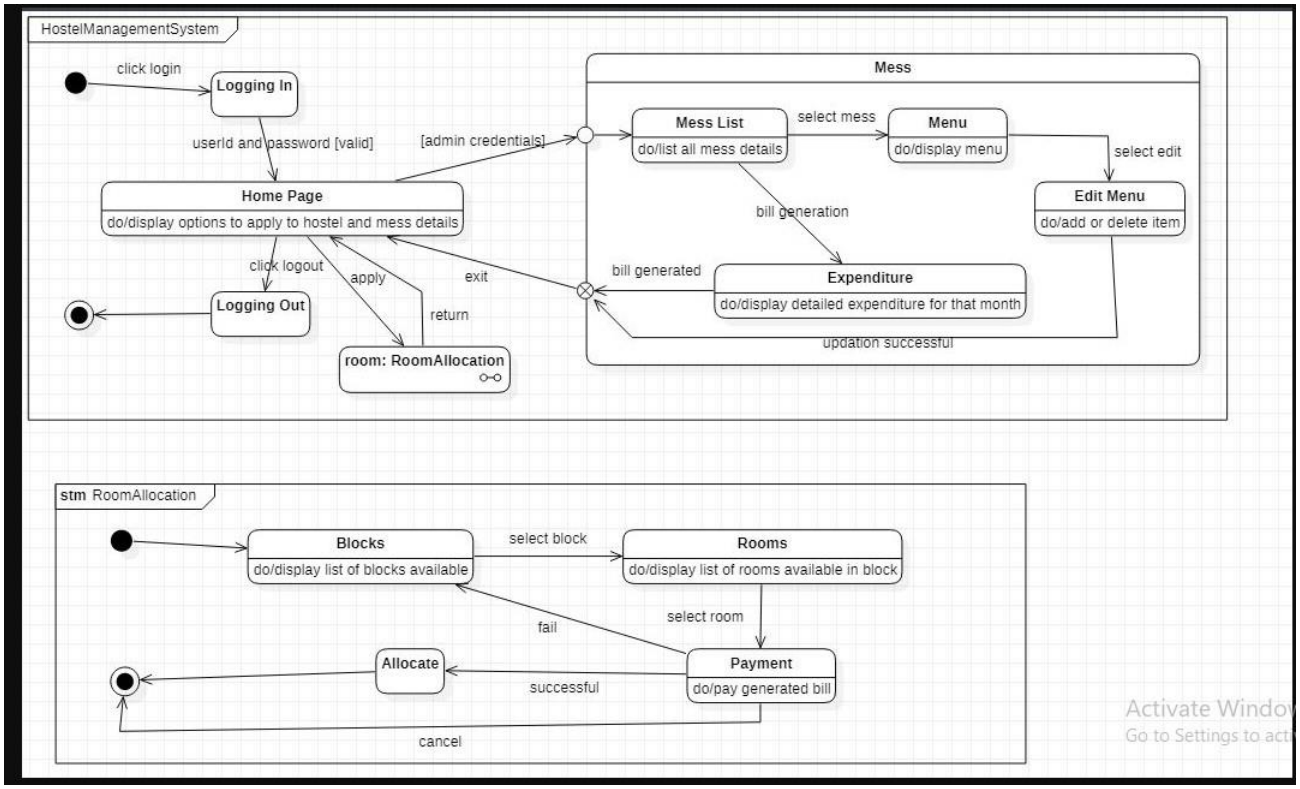
2. Advanced class diagram



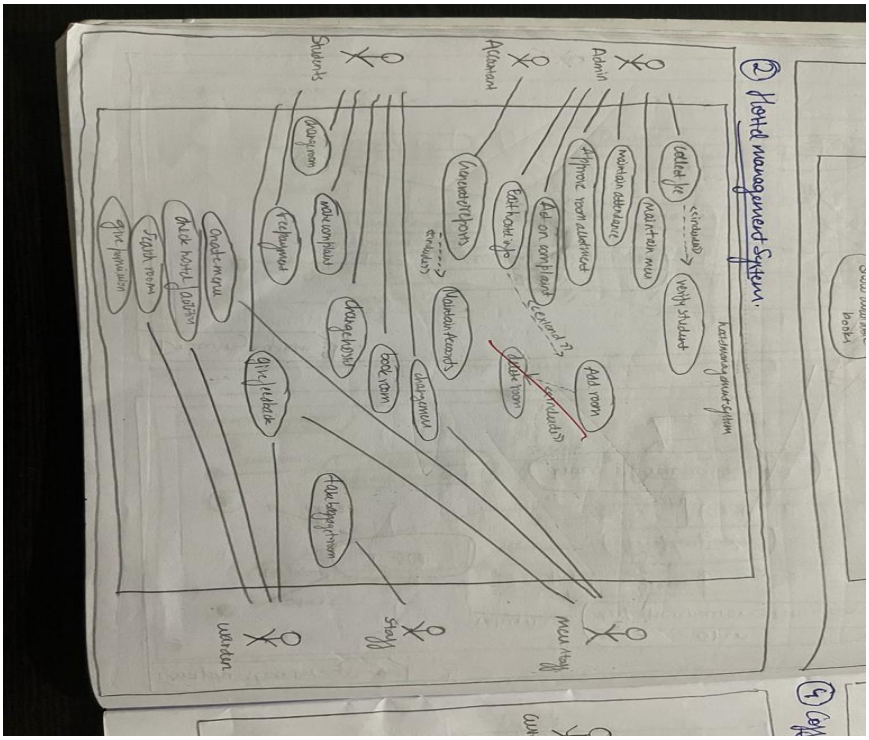

Person
+id
+name
+email
+update_profile()
+manages
Student
-student_id
-student_name
-room_no
-year
-age
-mobile_no
+get_student_info()
+set_student_info()
+room_allocation()
Administrator
+student_id
+room_id
+balance
+deposit
+duration
-receipt_generation()
-update_stud_info()
+update_room_info()
+manage_mess_canteen()
Warden
-building_no
+contact_no
+get_student_info()
+mark_attendance()
+maintain_utilities()
+stays in
+allocated
+decides room
+generates
Hostel
-hostel_id
-no_of_students
-hostel_name
+get_hostel_detaisl()
+update_hostel_details()
Room_allocation
-room_no
-no_of_students
-student_id
+room: RoomType
+update_room()
+room_allocation()
Receipt_generation
-receipt_id
-receipt_amount
-student_id
+receipt_generation()
«enumeration»
RoomType
singleRoomAC
doubleRoomAC
duplex
singleRoomNonAC
doubleRoomNonAC
Rooms
+room_no
+composed of
Bed
+bed_no

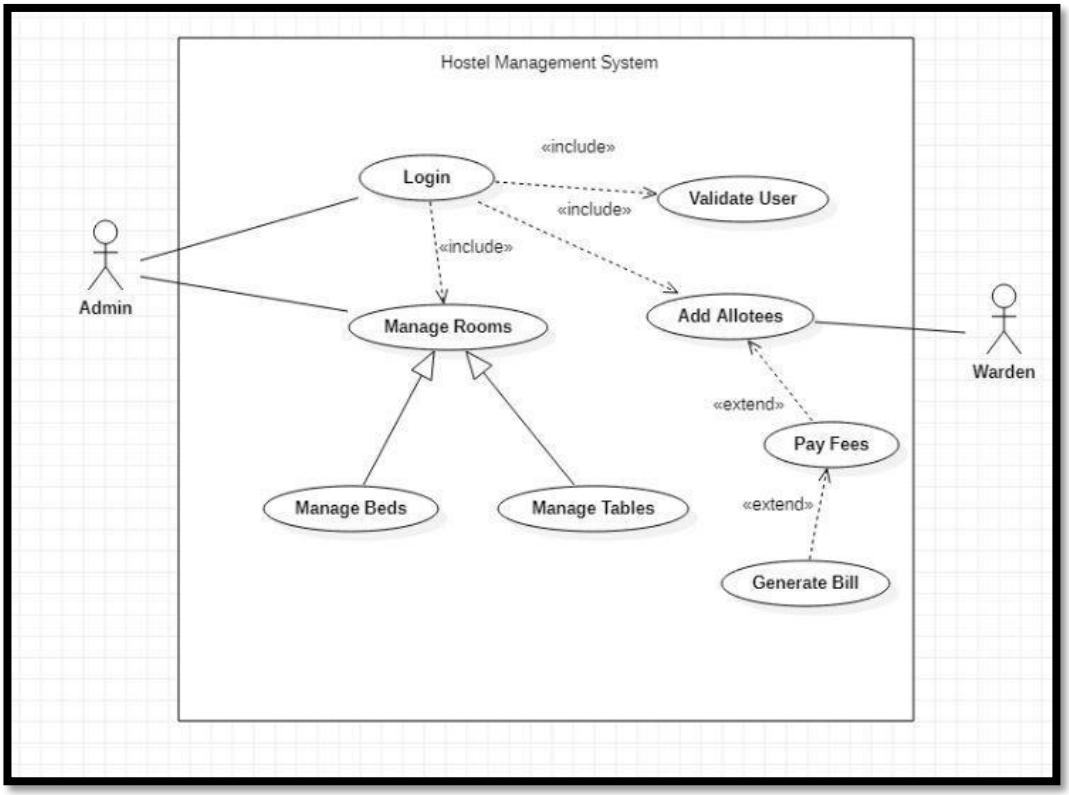
3. Advanced state diagram



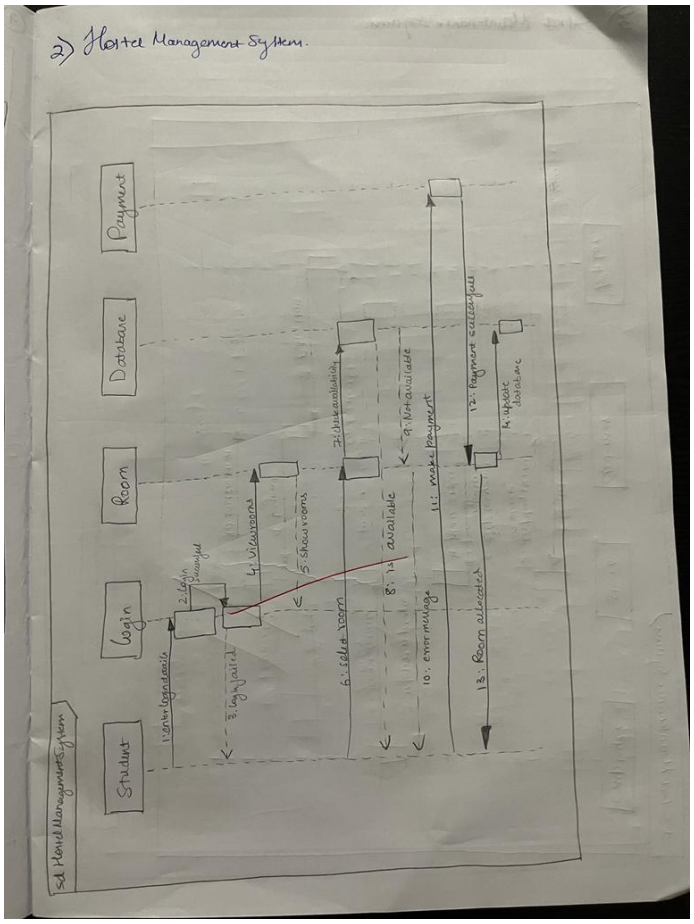


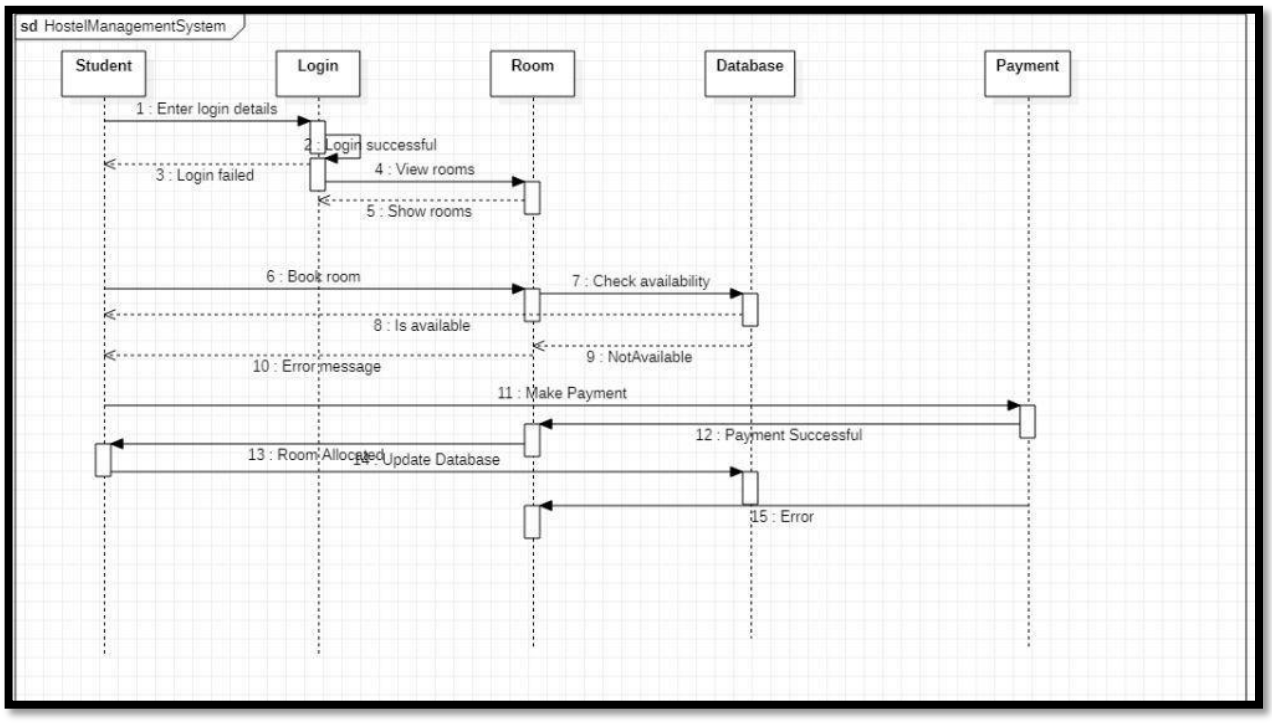
4. Advanced use case diagram



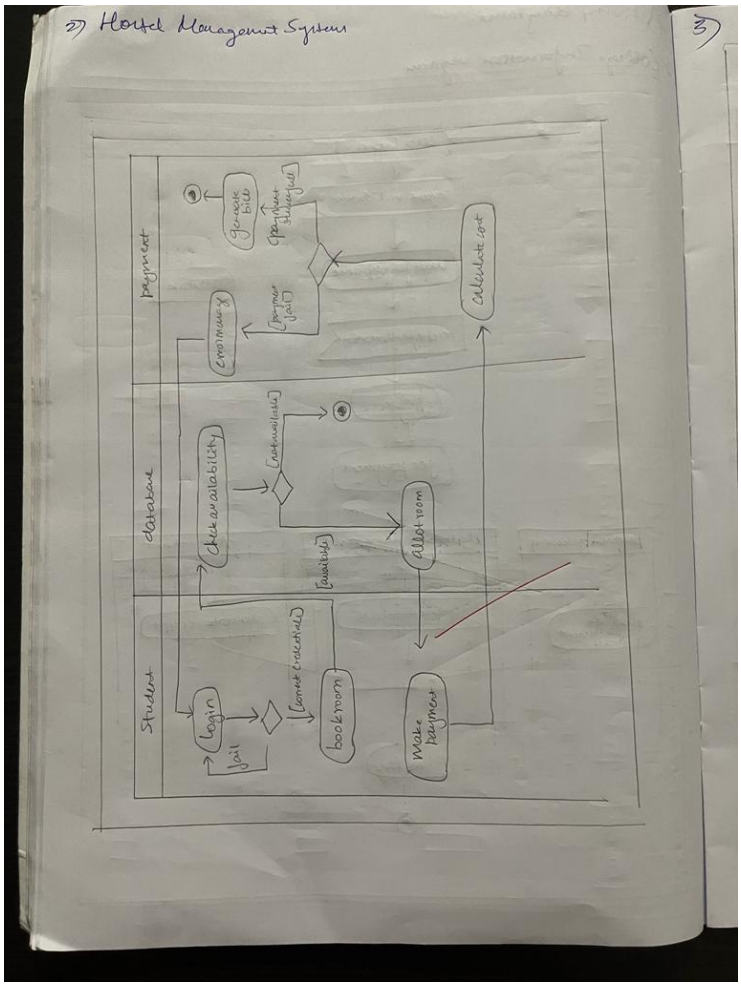


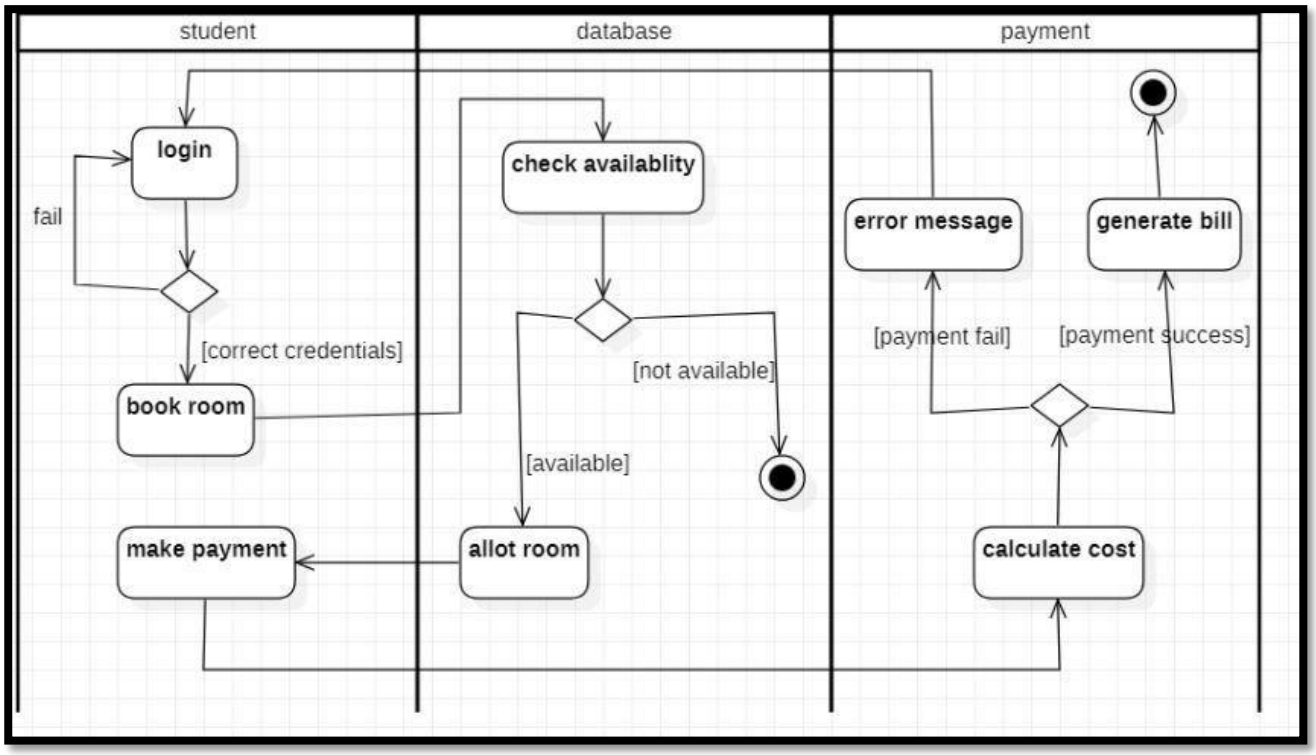
5. Advanced sequence diagram





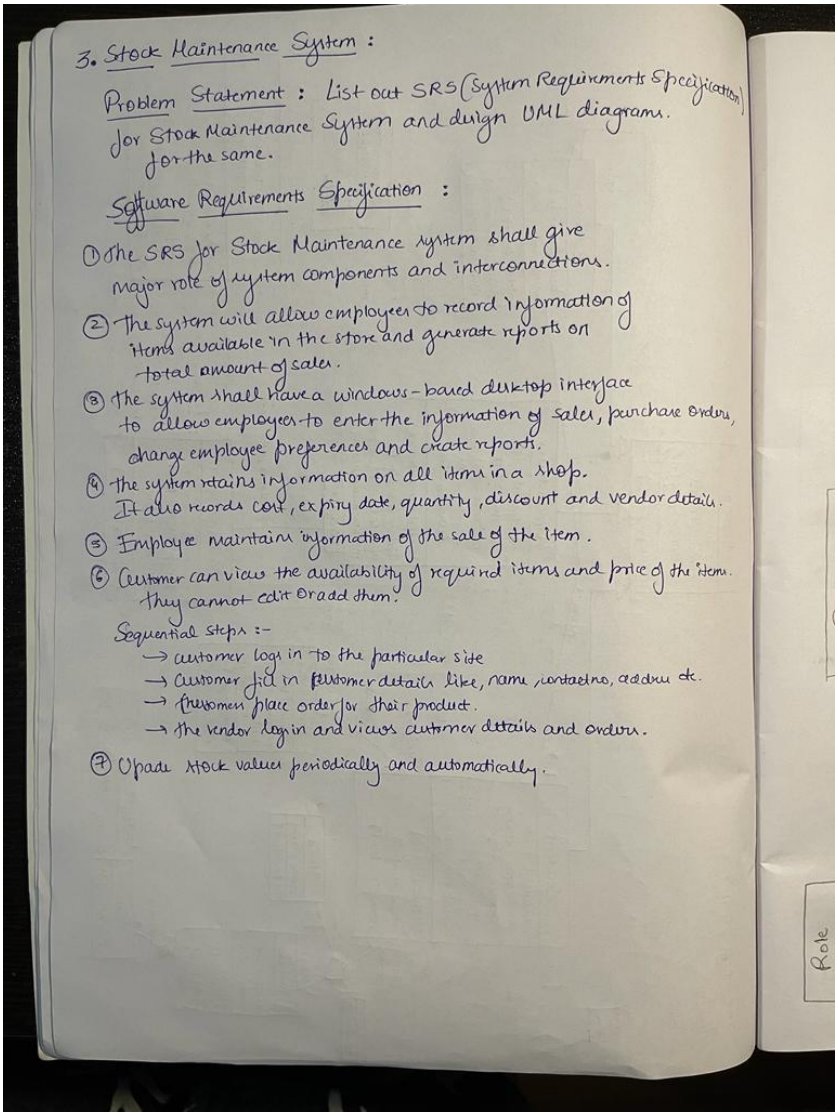
6. Advanced activity diagram





Exercise 3: Stock Management System

1. SRS



3. Stock Maintenance System :

Problem Statement : List out SRS (System Requirements Specification) for Stock Maintenance System and design UML diagrams for the same.

Software Requirements Specification :

① The SRS for Stock Maintenance system shall give major role of system components and interconnections.

② The system will allow employees to record information of items available in the store and generate reports on total amount of sales.

③ The system shall have a windows-based desktop interface to allow employees to enter the information of sales, purchase orders, change employee preferences and create reports.

④ The system retains information on all items in a shop. It also records cost, expiry date, quantity, discount and vendor details.

⑤ Employee maintain information of the sale of the item.

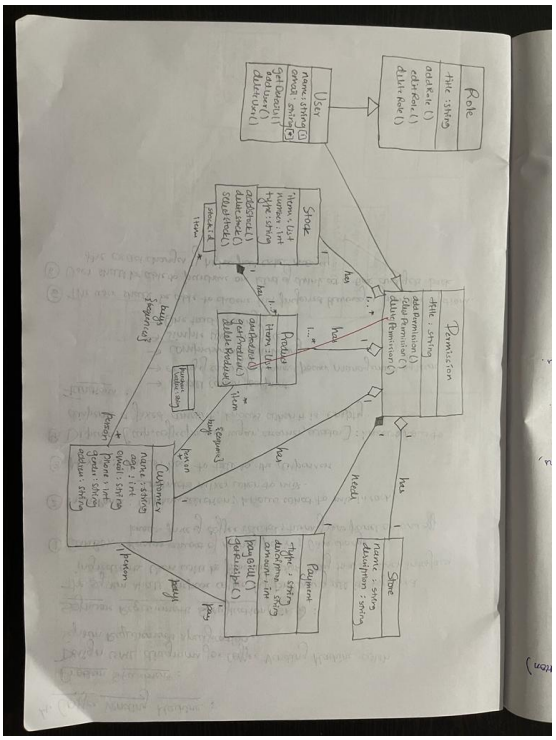
⑥ Customer can view the availability of required items and price of the items. They cannot edit or add them.

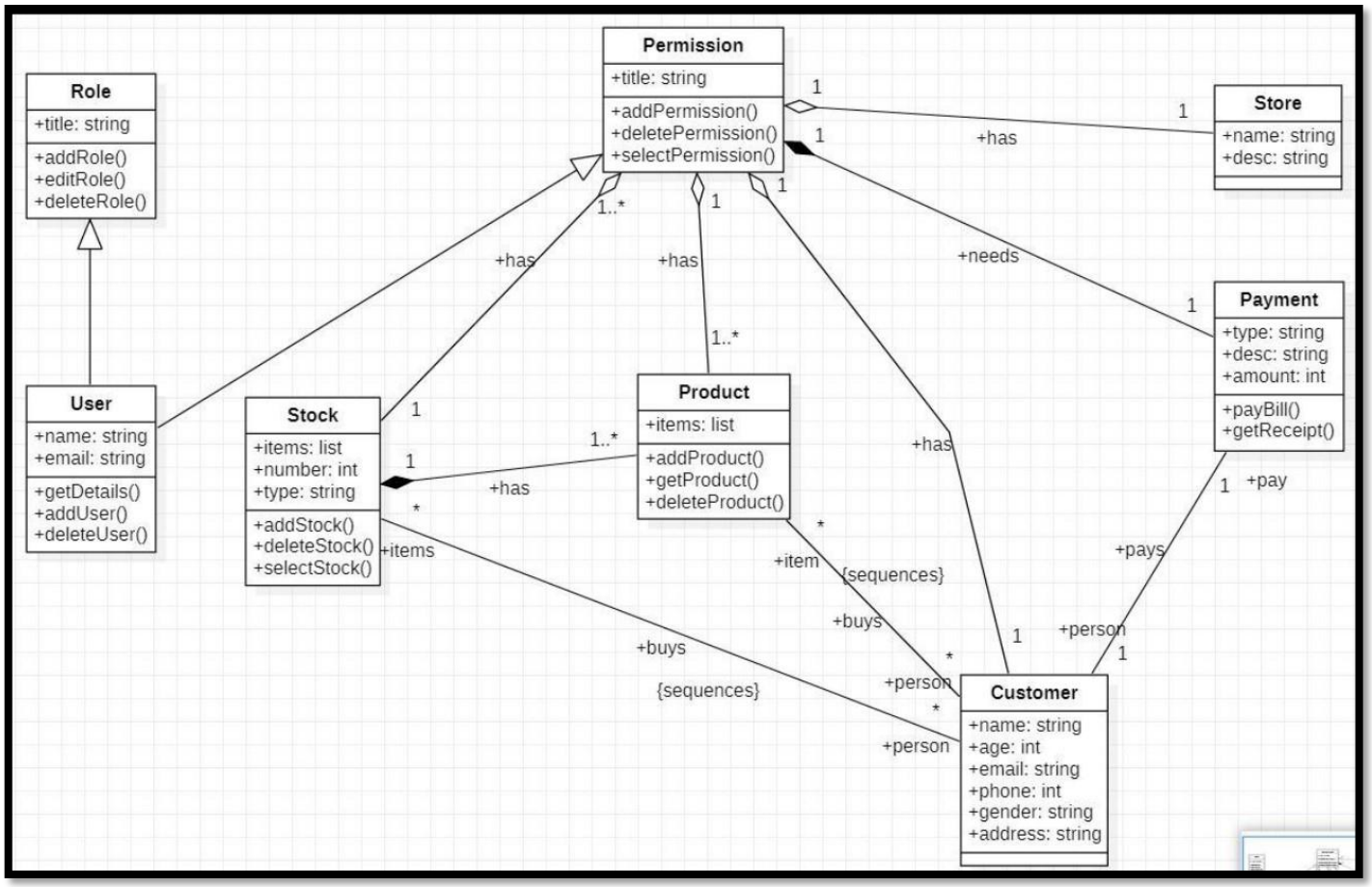
Sequential steps :-
- → customer login to the particular site
- → Customer fill in customer details like, name, contactno, address etc.
- → Customers place order for their product.
- → the vendor login and views customer details and orders.

⑦ Update stock values periodically and automatically.

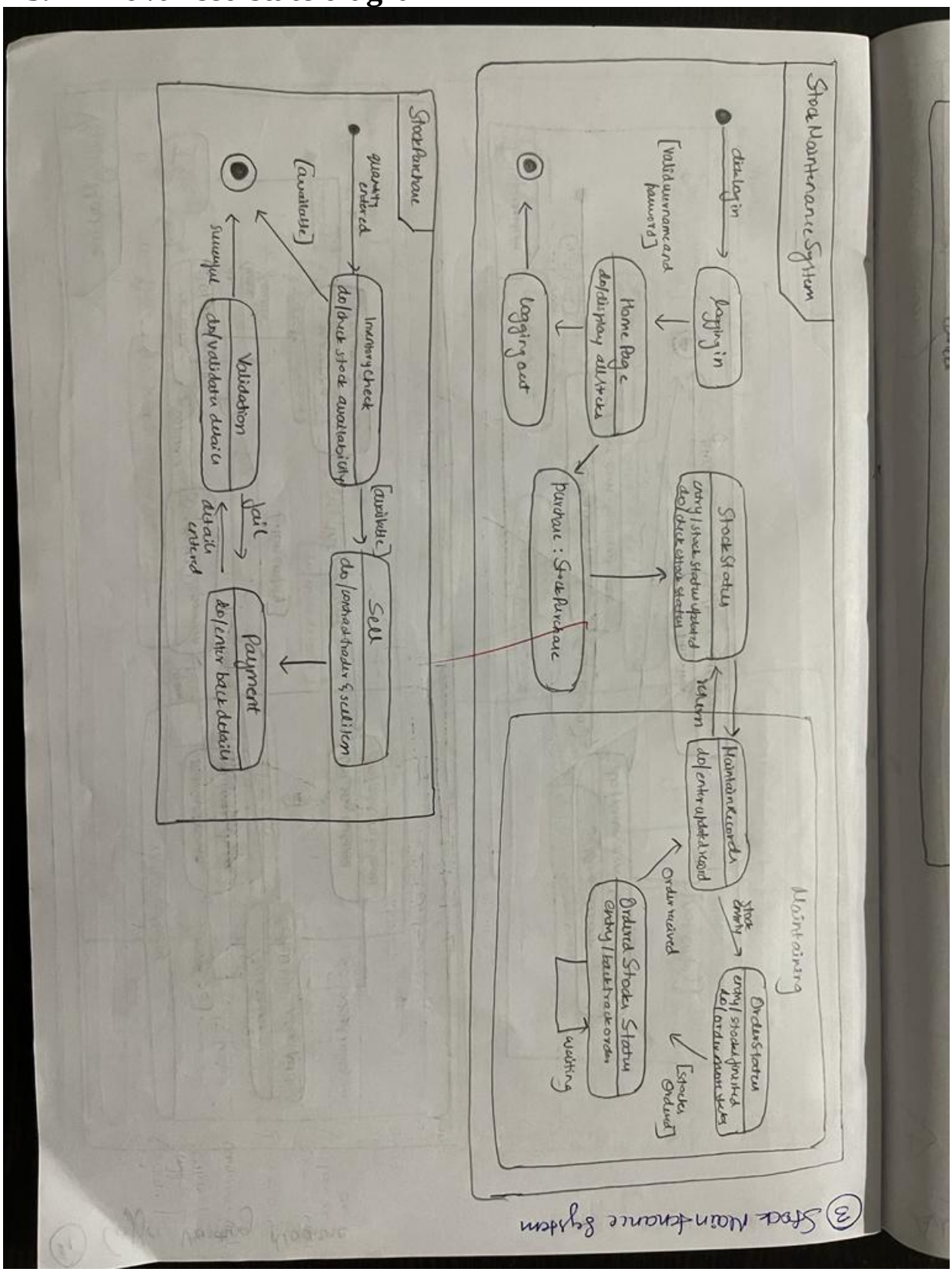
Role

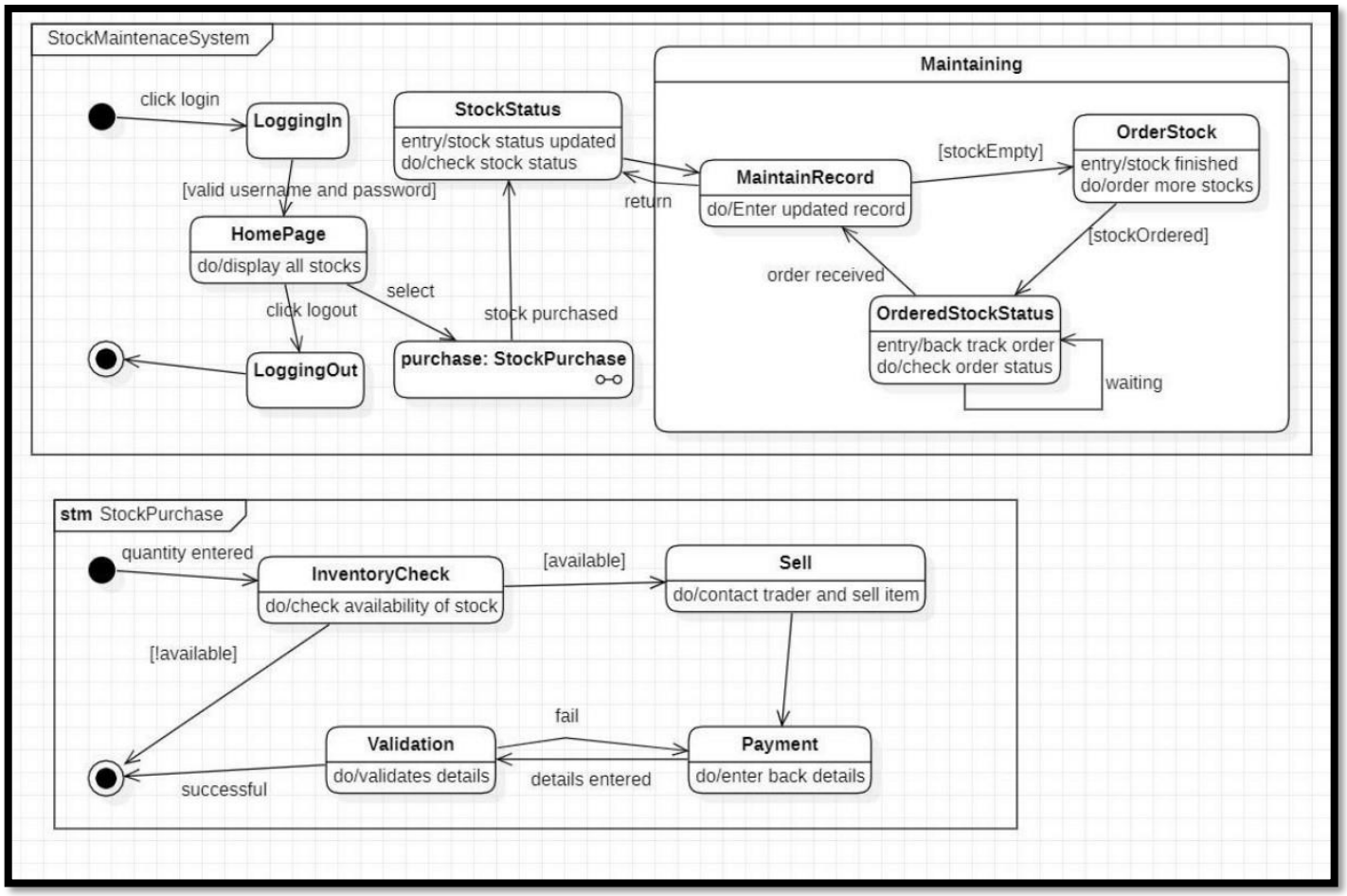
2. Advanced class diagram



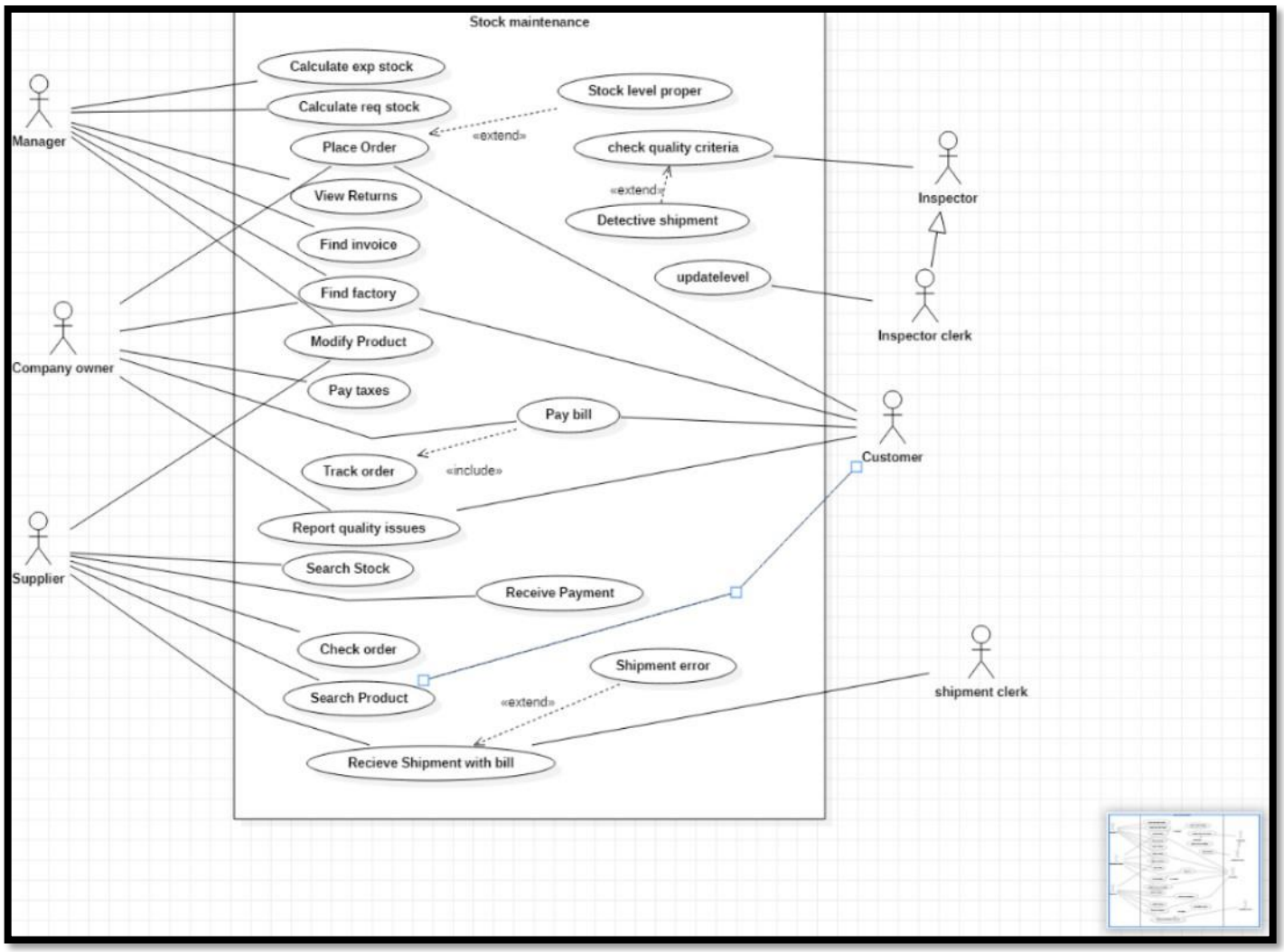


3. Advanced state diagram

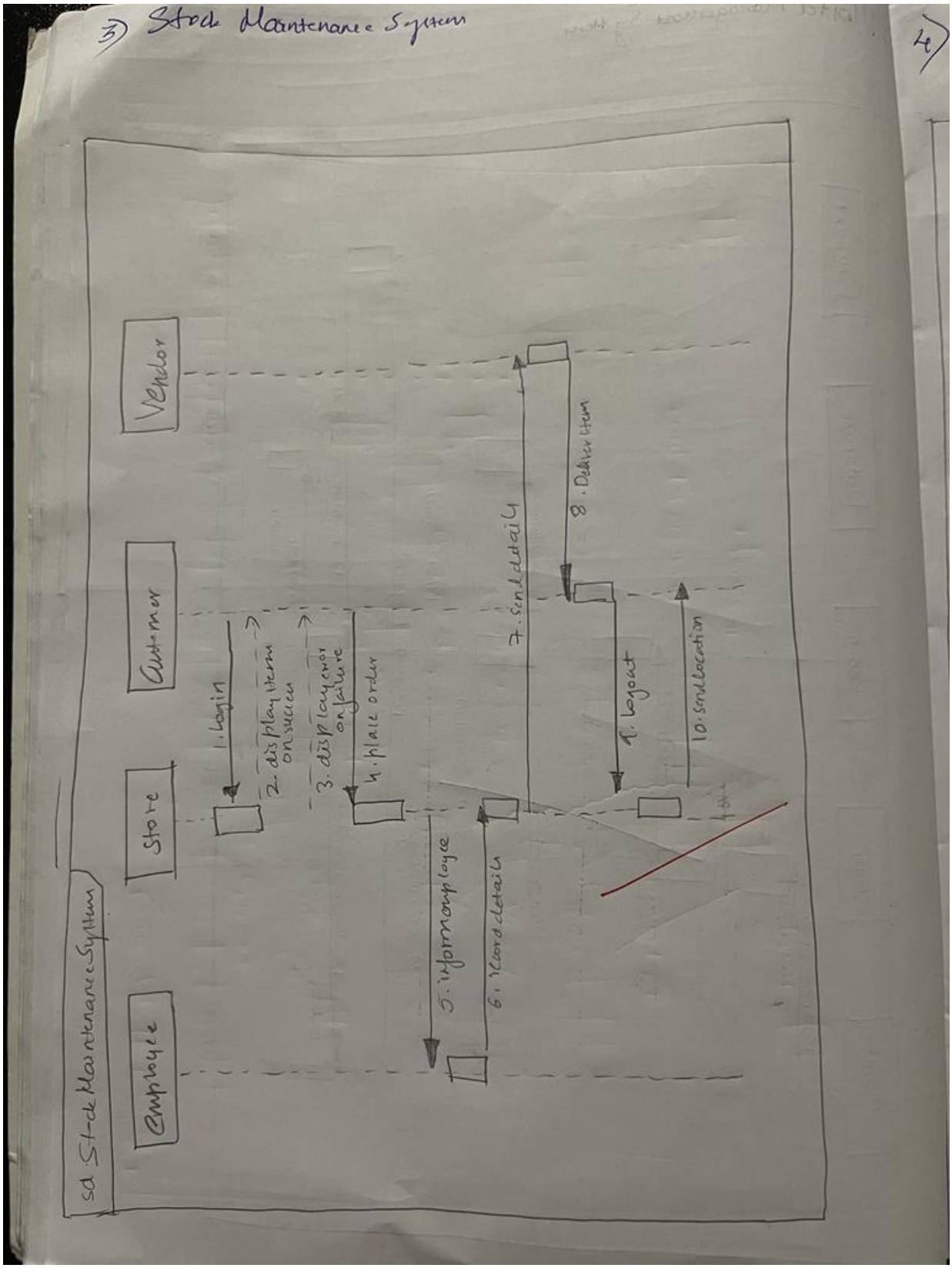


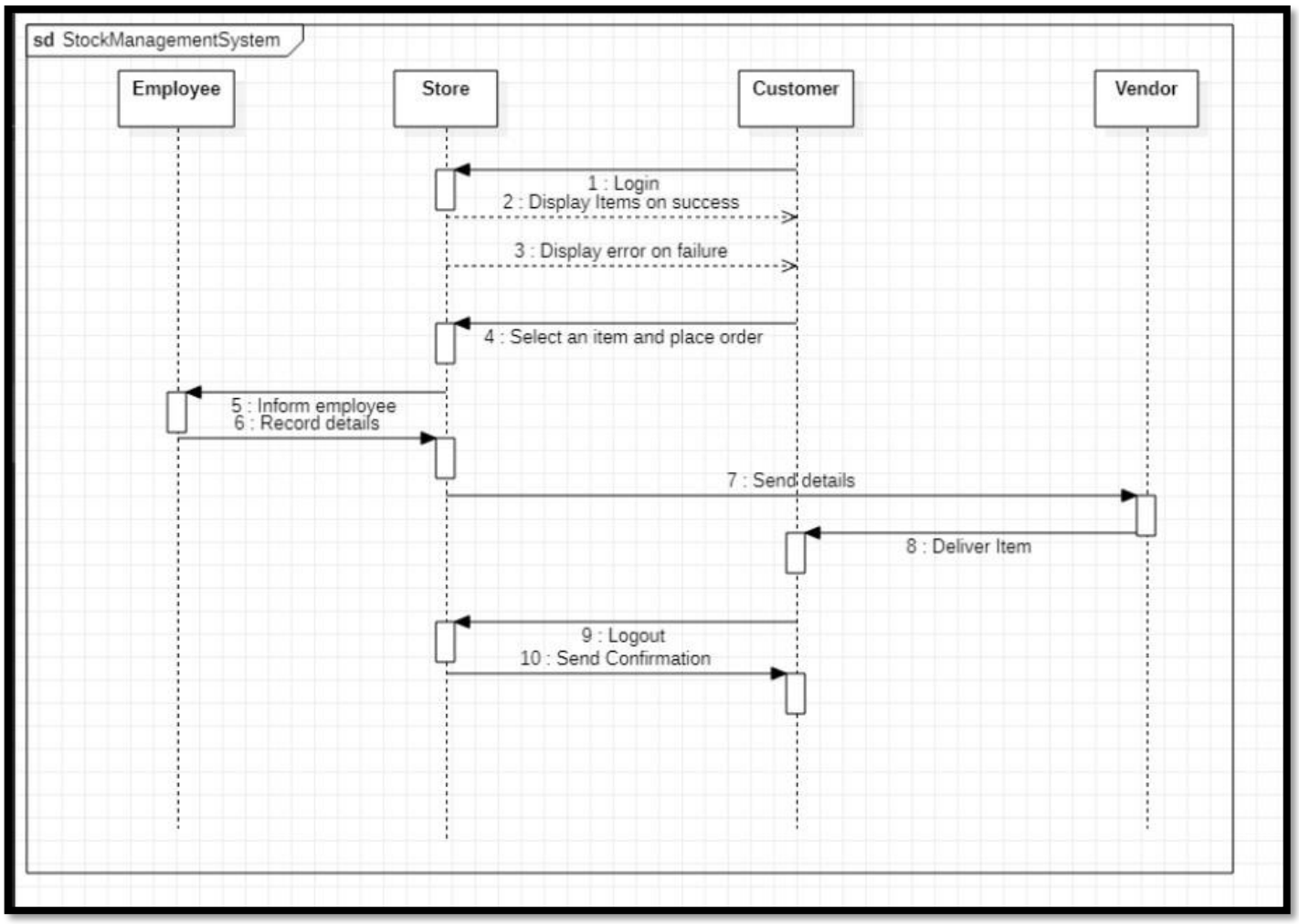

StockMaintenaceSystem
click login
LoggingIn
StockStatus
entry/stock status updated
do/check stock status
Maintaining
MaintainRecord
do/Enter updated record
[stockEmpty]
OrderStock
entry/stock finished
do/order more stocks
return
[valid username and password]
HomePage
do/display all stocks
[stockOrdered]
order received
OrderedStockStatus
entry/back track order
do/check order status
waiting
select
click logout
stock purchased
LoggingOut
purchase: StockPurchase
stm StockPurchase
quantity entered
InventoryCheck
do/check availability of stock
[available]
Sell
do/contact trader and sell item
[!available]
fail
Validation
do/validates details
Payment
do/enter back details
successful
details entered

4. Advanced use case diagram

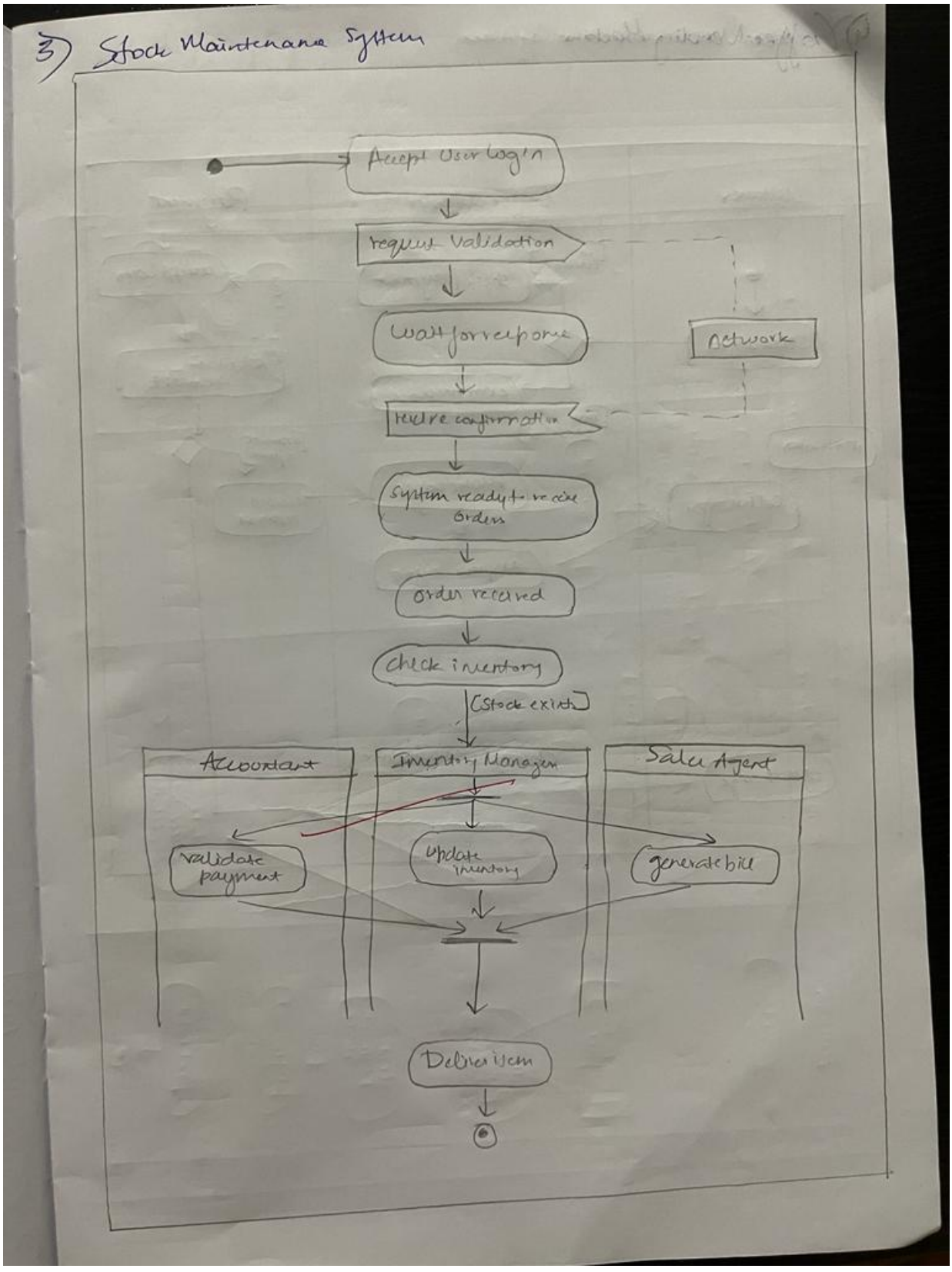

Stock maintenance
Calculate exp stock
Calculate req stock
Place Order
View Returns
Find invoice
Find factory
Modify Product
Pay taxes
Pay bill
Track order
Report quality issues
Search Stock
Receive Payment
Check order
Search Product
Recieve Shipment with bill
Stock level proper
«extend»
check quality criteria
«extend»
Detective shipment
updatelevel
«include»
«extend»
Manager
Company owner
Supplier
Inspector
Inspector clerk
Customer
shipment clerk

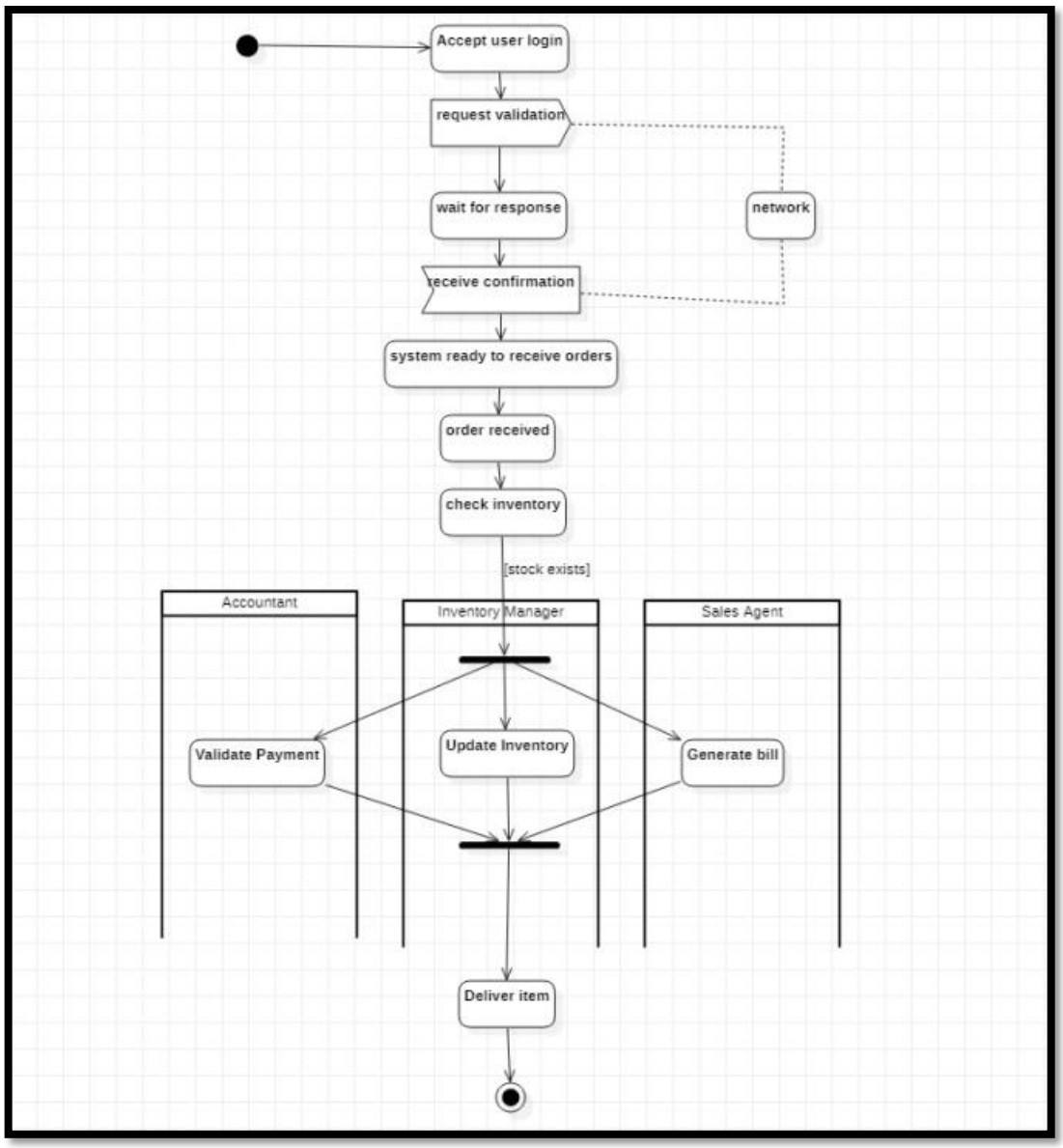
5. Advanced sequence diagram




sd StockManagementSystem
Employee
Store
Customer
Vendor
1 : Login
2 : Display Items on success
3 : Display error on failure
4 : Select an item and place order
5 : Inform employee
6 : Record details
7 : Send details
8 : Deliver Item
9 : Logout
10 : Send Confirmation

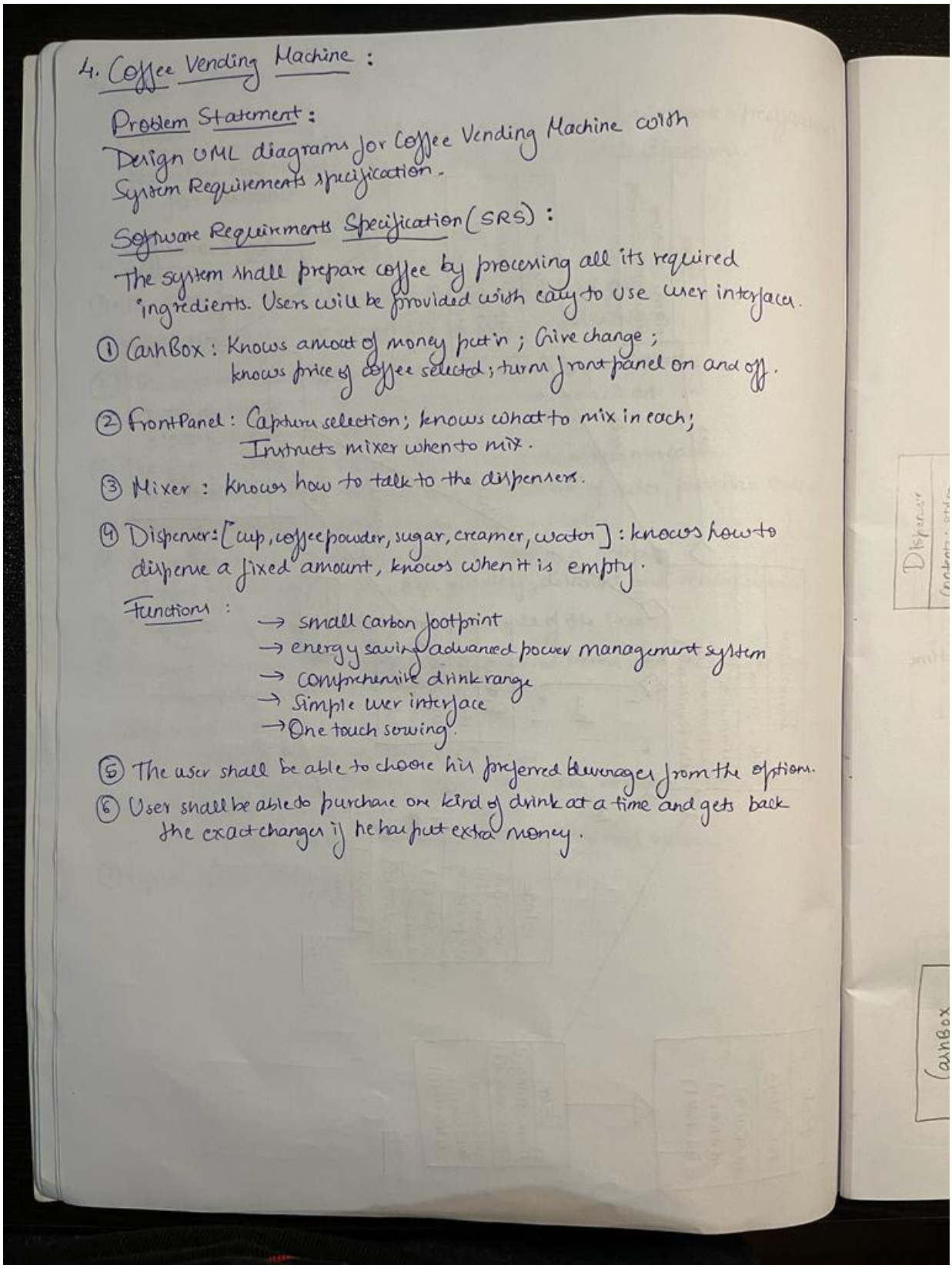
6. Advanced activity diagram




Accept user login
request validation
wait for response
network
receive confirmation
system ready to receive orders
order received
check inventory
[stock exists]
Accountant
Inventory Manager
Sales Agent
Validate Payment
Update Inventory
Generate bill
Deliver item

Exercise 4: Coffee Vending Machine

1. SRS



4. Coffee Vending Machine :

Problem Statement :
Design UML diagrams for Coffee Vending Machine with System Requirements specification.

Software Requirements Specification (SRS) :
The system shall prepare coffee by procesing all its required ingredients. Users will be provided with easy to use user interfaces.

① CashBox : Knows amount of money put in ; Give change ; knows price of coffee selected ; turns front panel on and off.

② FrontPanel : Capture selection ; knows what to mix in each ; Instructs mixer when to mix.

③ Mixer : knows how to talk to the dispensers.

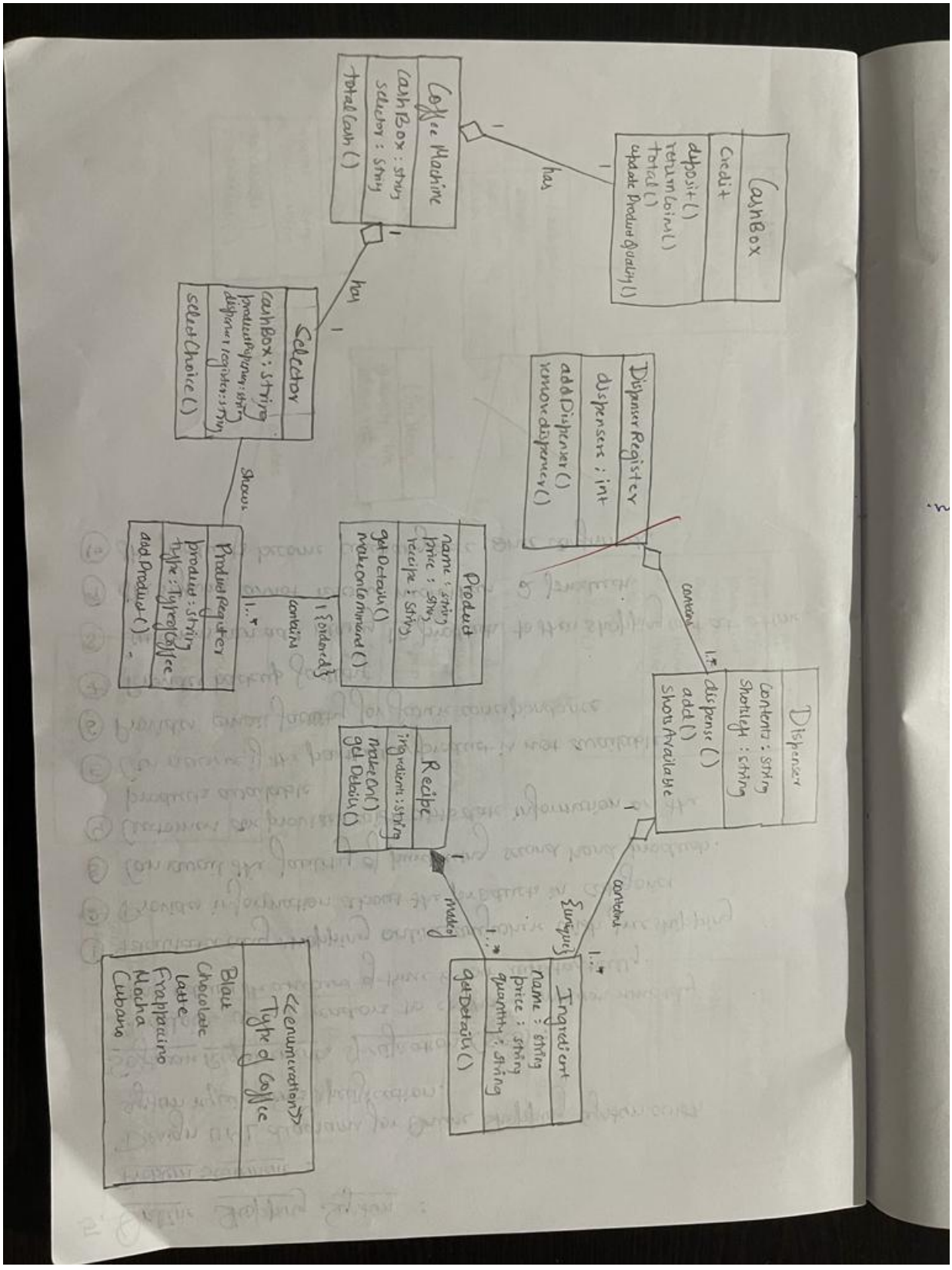
④ Dispenser : [cup, coffeepowder, sugar, creamer, water] : knows how to dispense a fixed amount, knows when it is empty.

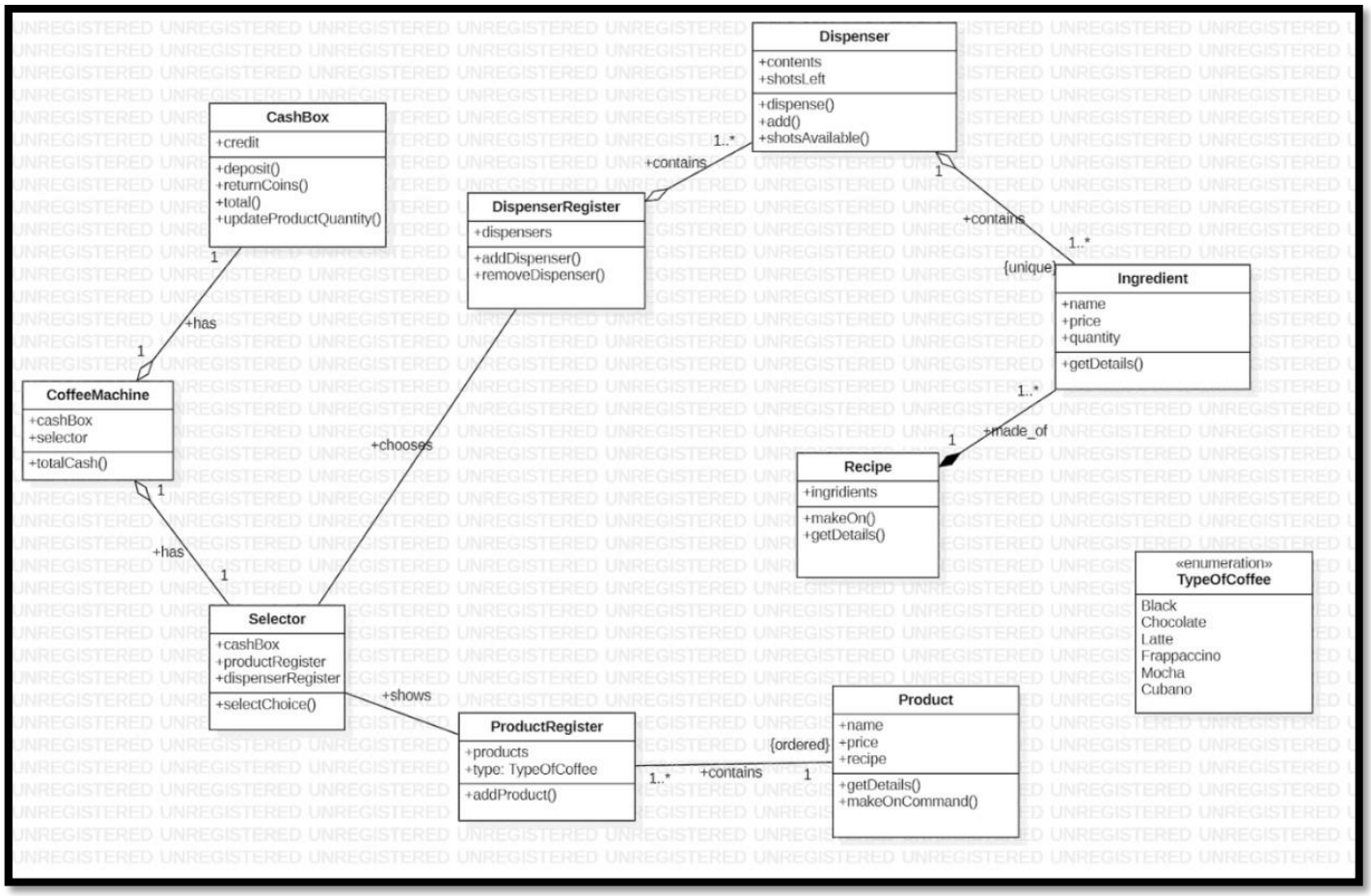
Functions :
→ small carbon footprint
→ energy savin advanced power management system
→ comprehensive drink range
→ Simple user interface
→ One touch sowing.

⑤ The user shall be able to choose his preferred beverages from the options.

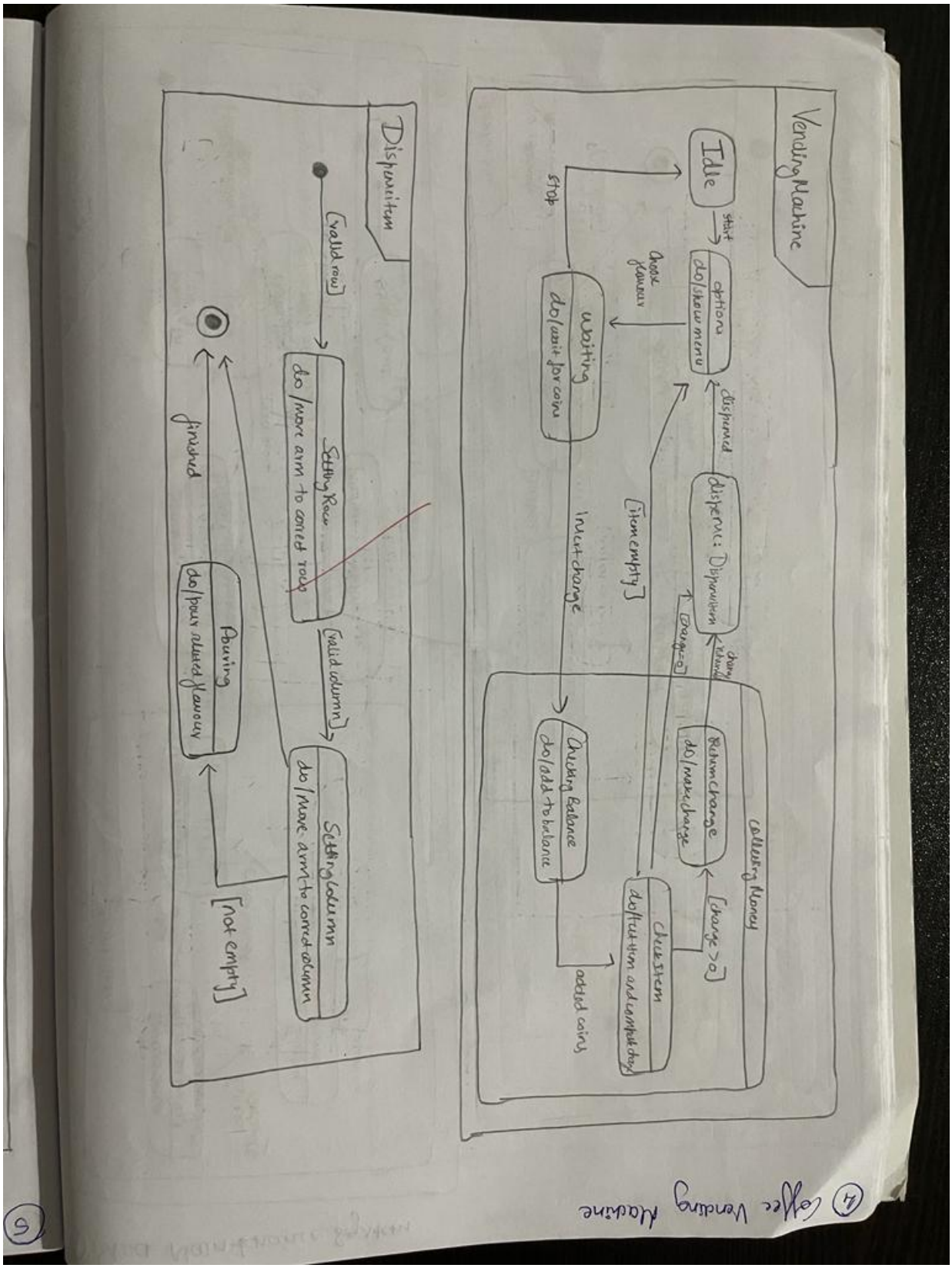
⑥ User shall be able to purchase one kind of drink at a time and gets back the exact changes if he has put extra money.

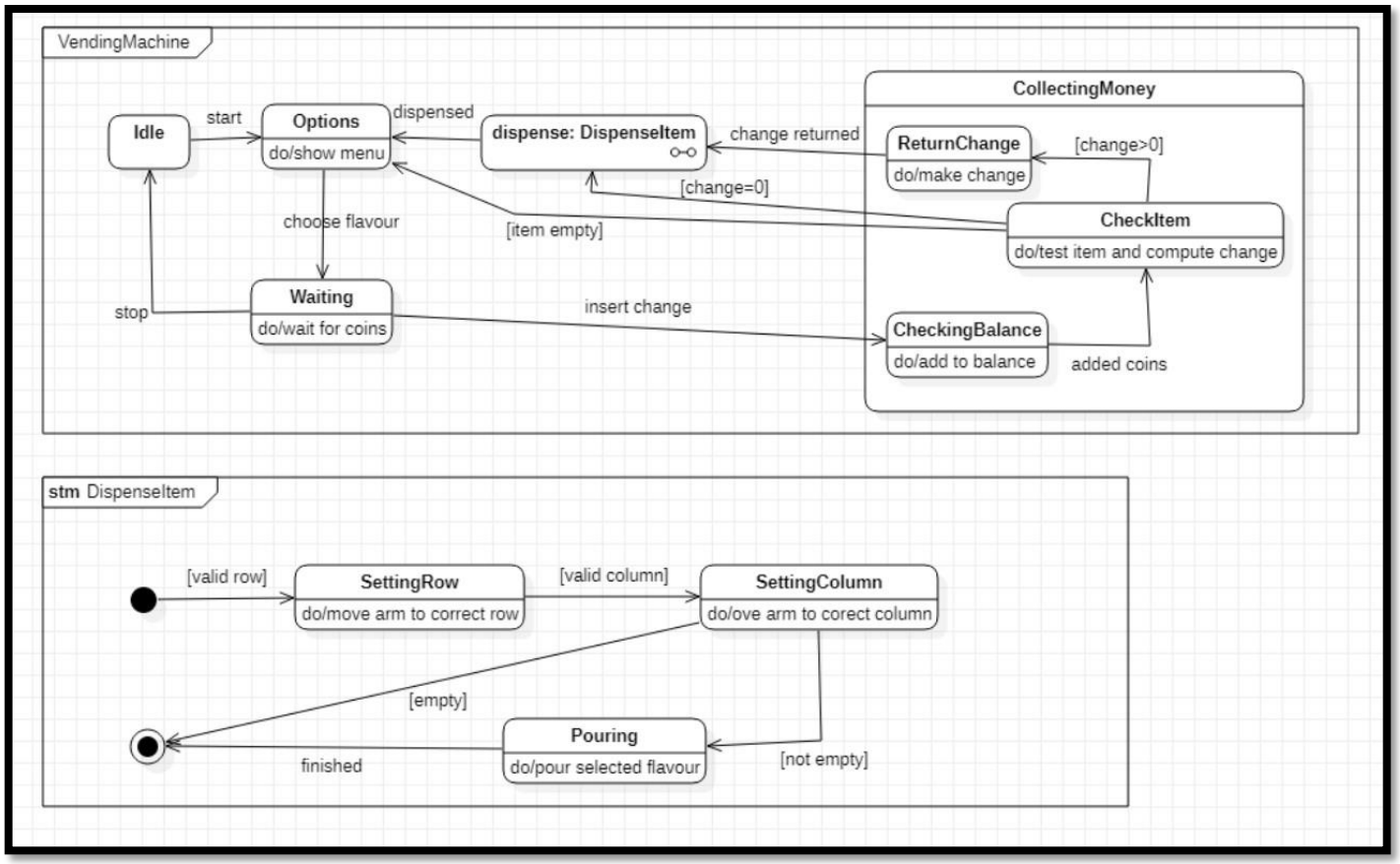
2. Advanced class diagram



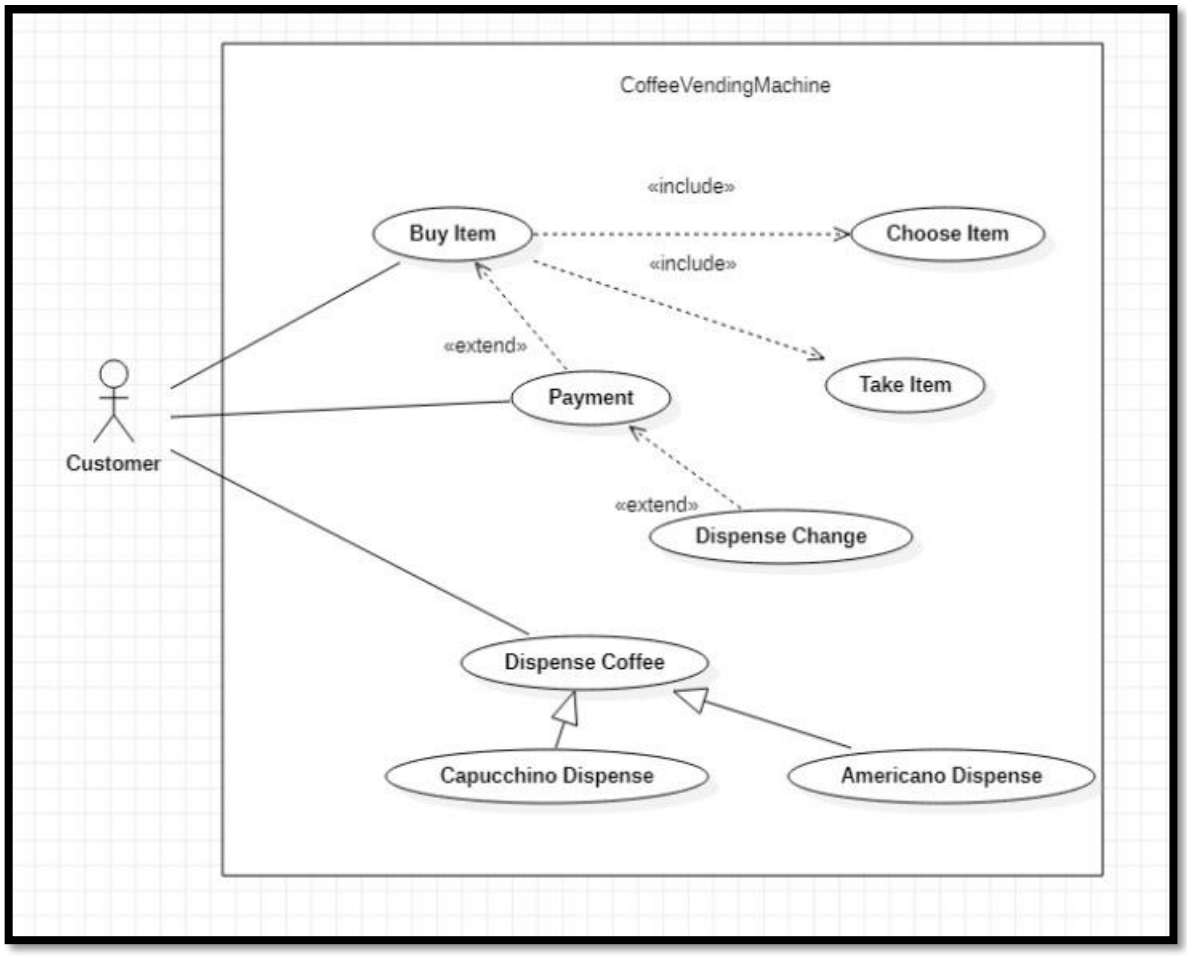

Dispenser
+contents
+shotsLeft
+dispense()
+add()
+shotsAvailable()
CashBox
+credit
+deposit()
+returnCoins()
+total()
+updateProductQuantity()
DispenserRegister
+dispensers
+addDispenser()
+removeDispenser()
Ingredient
+name
+price
+quantity
+getDetails()
CoffeeMachine
+cashBox
+selector
+totalCash()
Recipe
+ingridients
+makeOn()
+getDetails()
«enumeration»
TypeOfCoffee
Black
Chocolate
Latte
Frappaccino
Mocha
Cubano
Selector
+cashBox
+productRegister
+dispenserRegister
+selectChoice()
Product
+name
+price
+recipe
+getDetails()
+makeOnCommand()
ProductRegister
+products
+type: TypeOfCoffee
+addProduct()
1..*
+contains
1
+contains
1..*
{unique}
1
+has
1
1..*
+made_of
1
+has
1
+chooses
+shows
1..*
+contains
{ordered}
1

3. Advanced state diagram

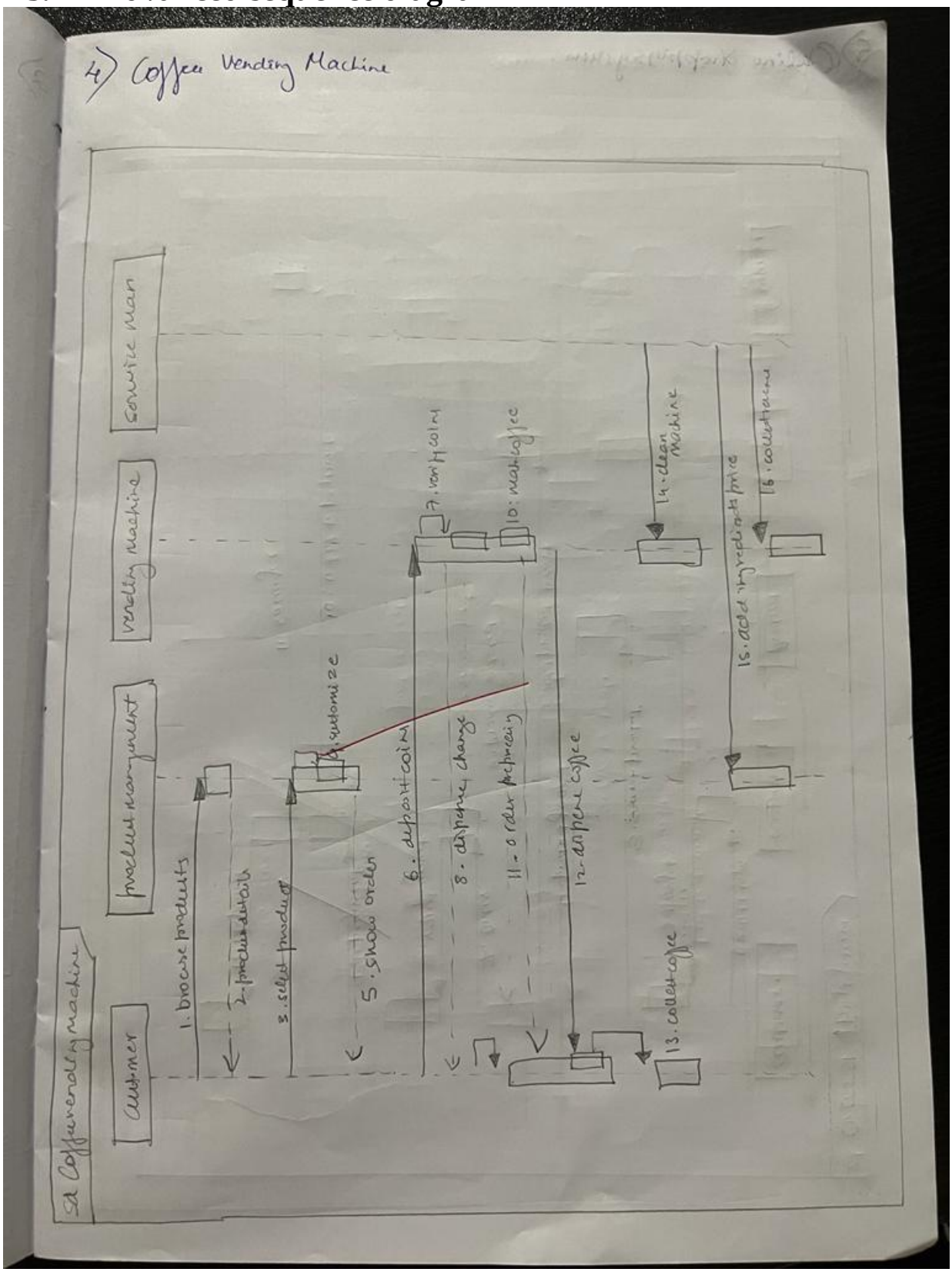


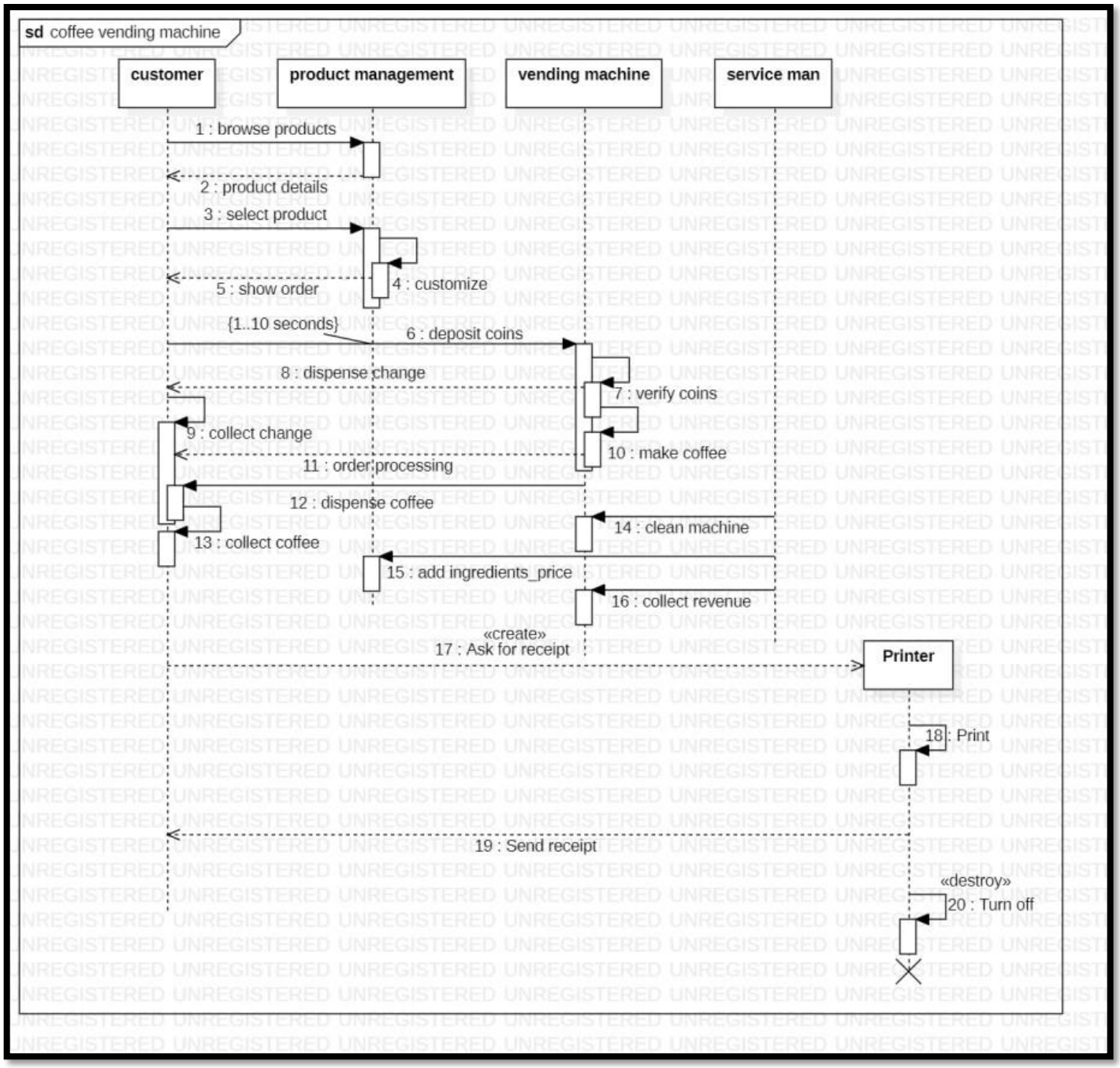

VendingMachine
Idle
start
Options
do/show menu
dispensed
dispense: DispenseItem
change returned
CollectingMoney
ReturnChange
do/make change
[change>0]
[change=0]
choose flavour
[item empty]
CheckItem
do/test item and compute change
stop
Waiting
do/wait for coins
insert change
CheckingBalance
do/add to 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]

4. Advanced use case diagram

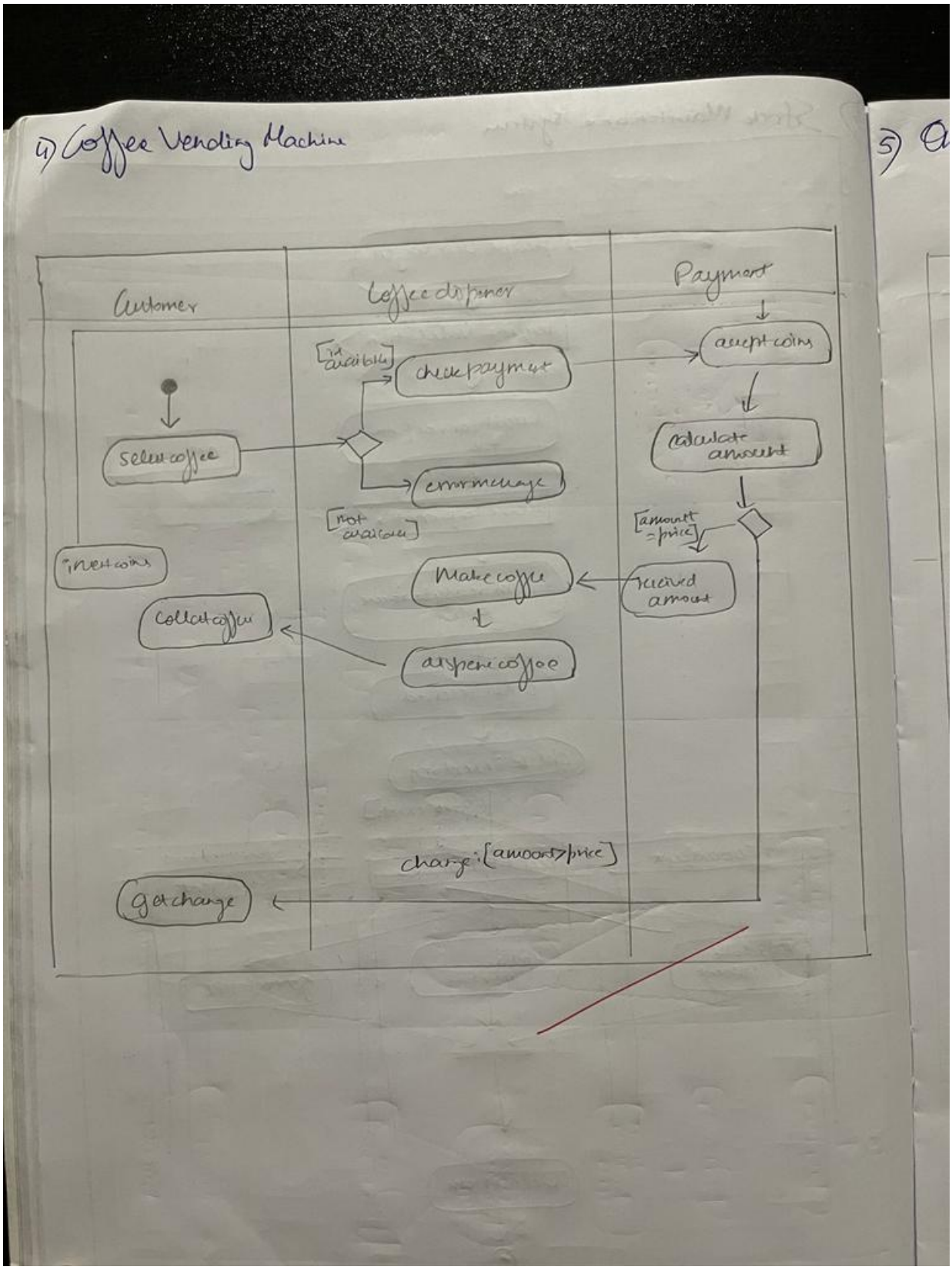

CoffeeVendingMachine
«include»
Buy Item
Choose Item
«include»
«extend»
Take Item
Payment
Customer
«extend»
Dispense Change
Dispense Coffee
Capucchino Dispense
Americano Dispense

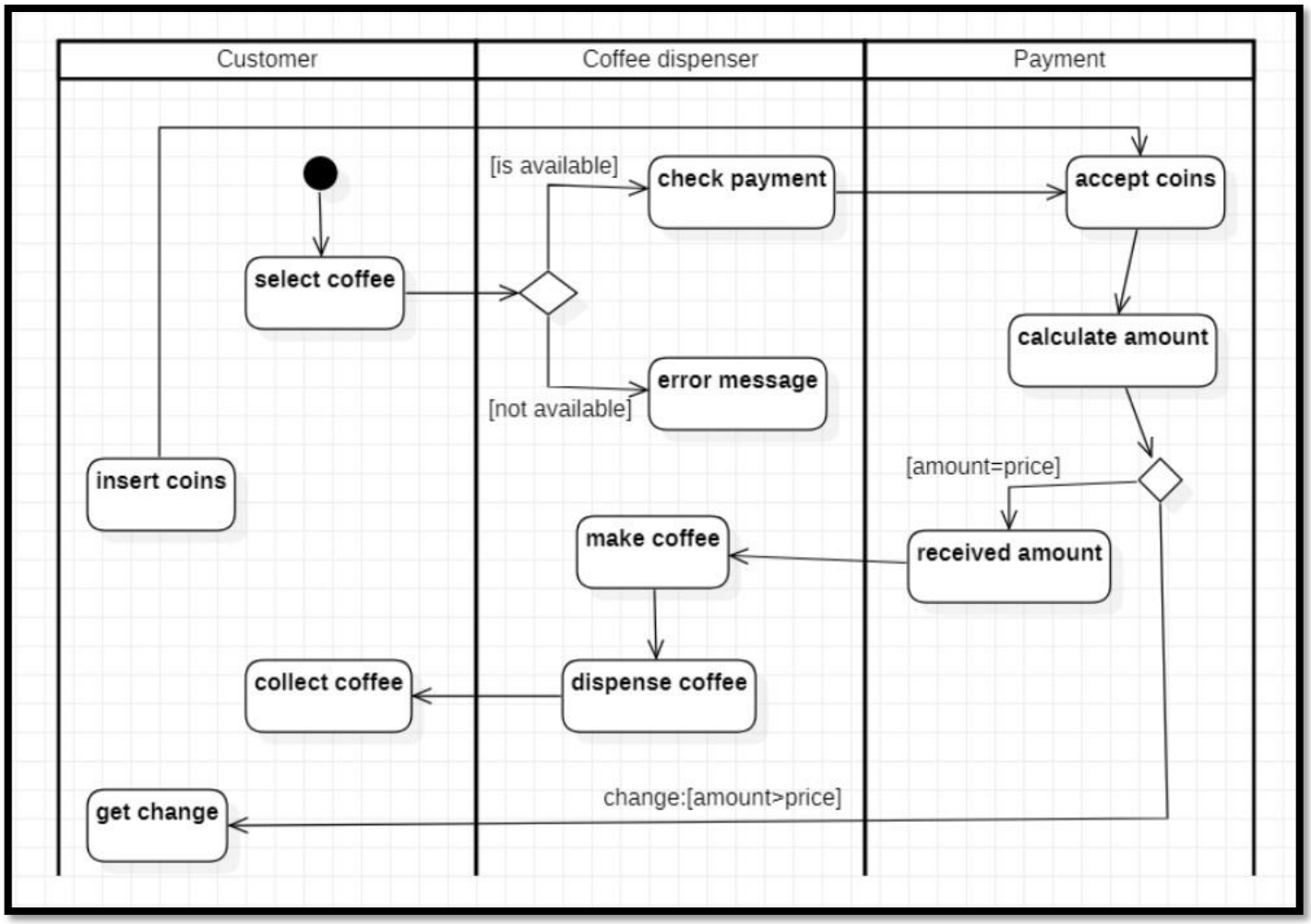
5. Advanced sequence diagram




sd coffee vending machine
customer
product management
vending machine
service man
1 : browse products
2 : product details
3 : select product
4 : customize
5 : show order
{1..10 seconds}
6 : deposit coins
7 : verify coins
8 : dispense change
9 : collect change
10 : make coffee
11 : order processing
12 : dispense coffee
13 : collect coffee
14 : clean machine
15 : add ingredients_price
16 : collect revenue
«create»
17 : Ask for receipt
Printer
18 : Print
19 : Send receipt
«destroy»
20 : Turn off

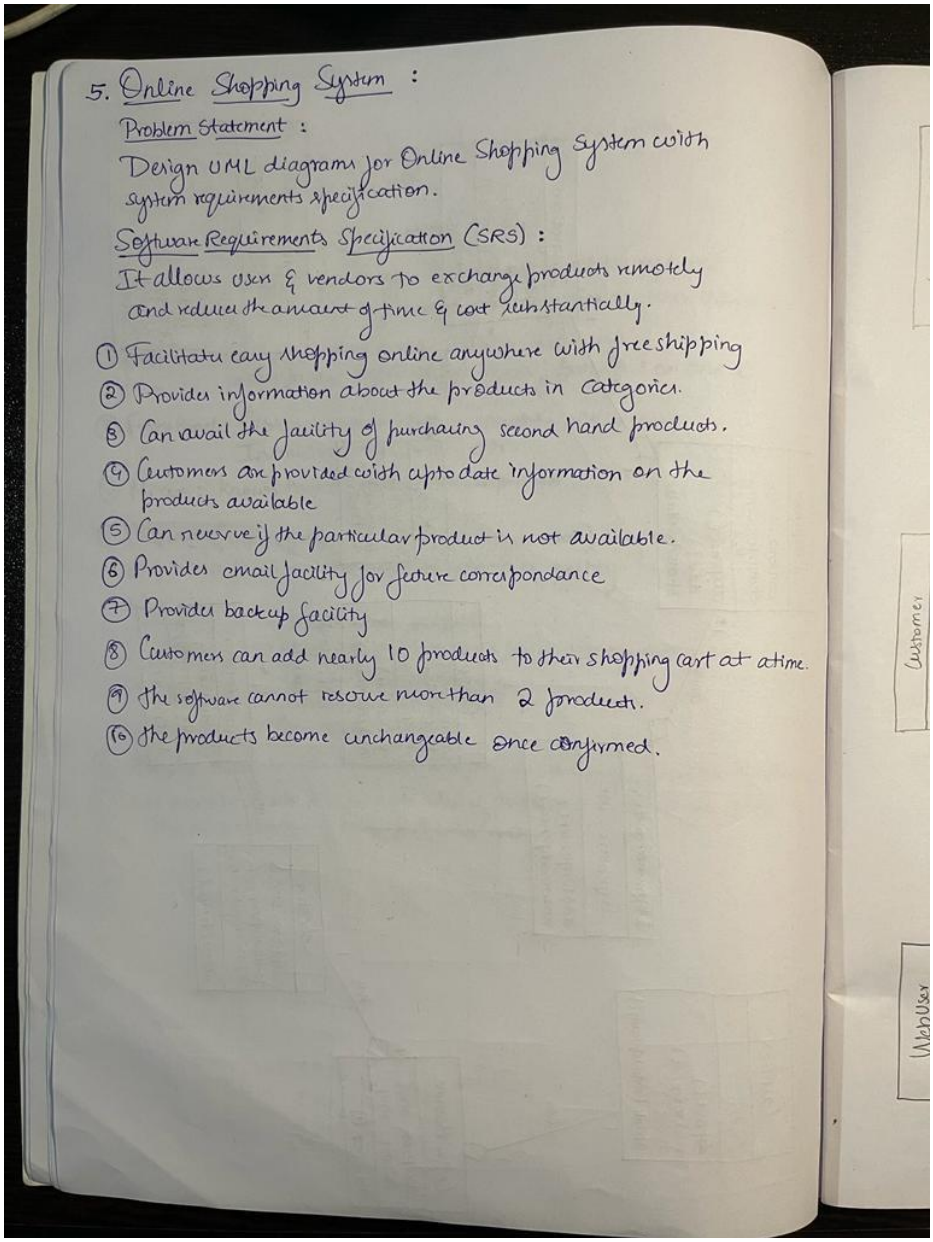
6. Advanced activity diagram




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

Exercise 5: Online Shopping System

1. SRS



5. Online Shopping System :

Problem Statement :

Design UML diagrams for Online Shopping System with system requirements specification.

Software Requirements Specification (SRS) :

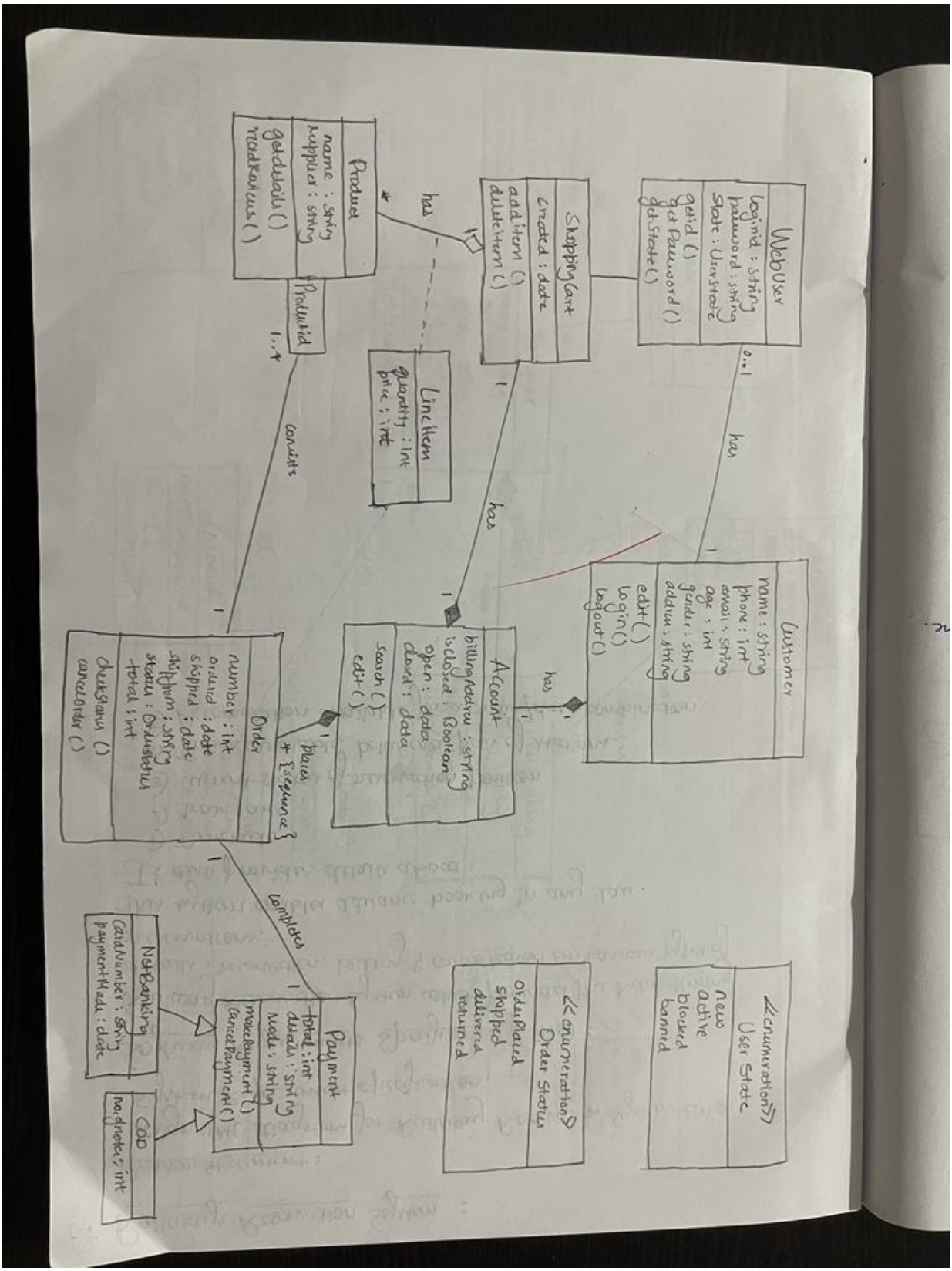
It allows user & vendors to exchange products remotely and reduces the amount of time & cost substantially.

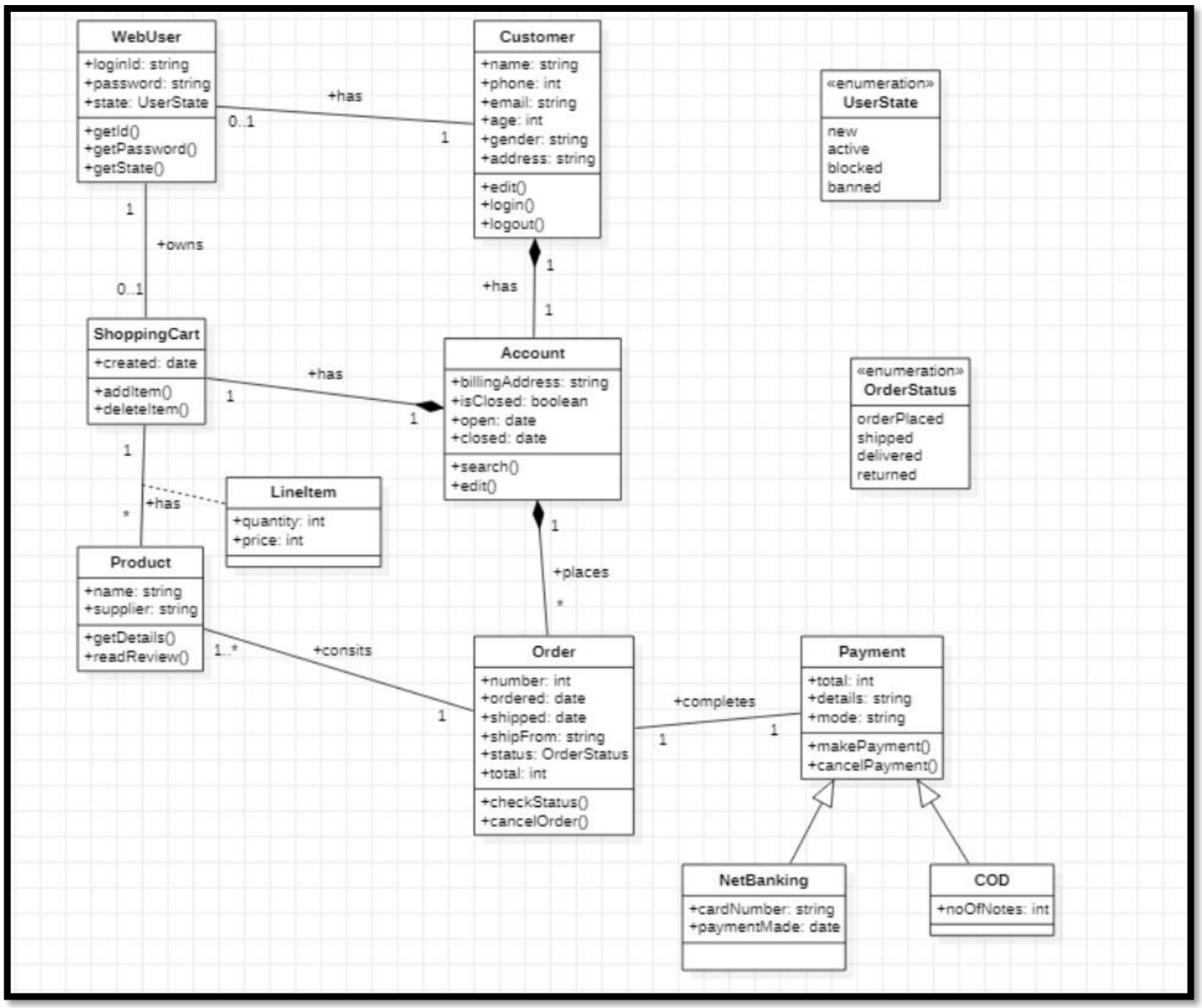
1. Facilitates easy shopping online anywhere with free shipping
2. Provides information about the products in categories.
3. Can avail the facility of purchasing second hand products.
4. Customers are provided with upto date information on the products available
5. Can reserve if the particular product is not available.
6. Provides email facility for secure correspondance
7. Provides backup facility
8. Customers can add nearly 10 products to their shopping cart at a time.
9. The software cannot reserve more than 2 products.
10. The products become unchangable once confirmed.

Customer

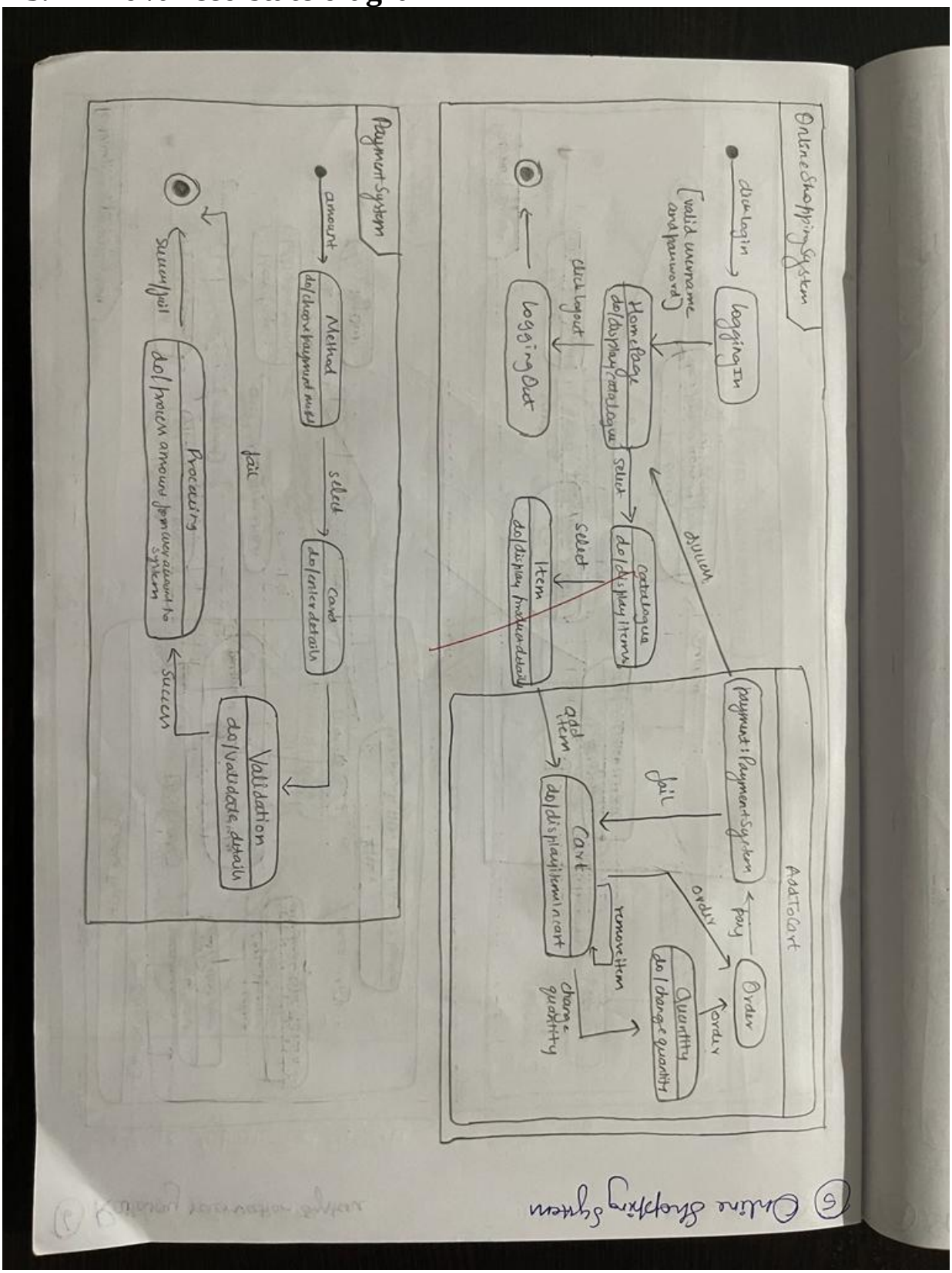
WebUser

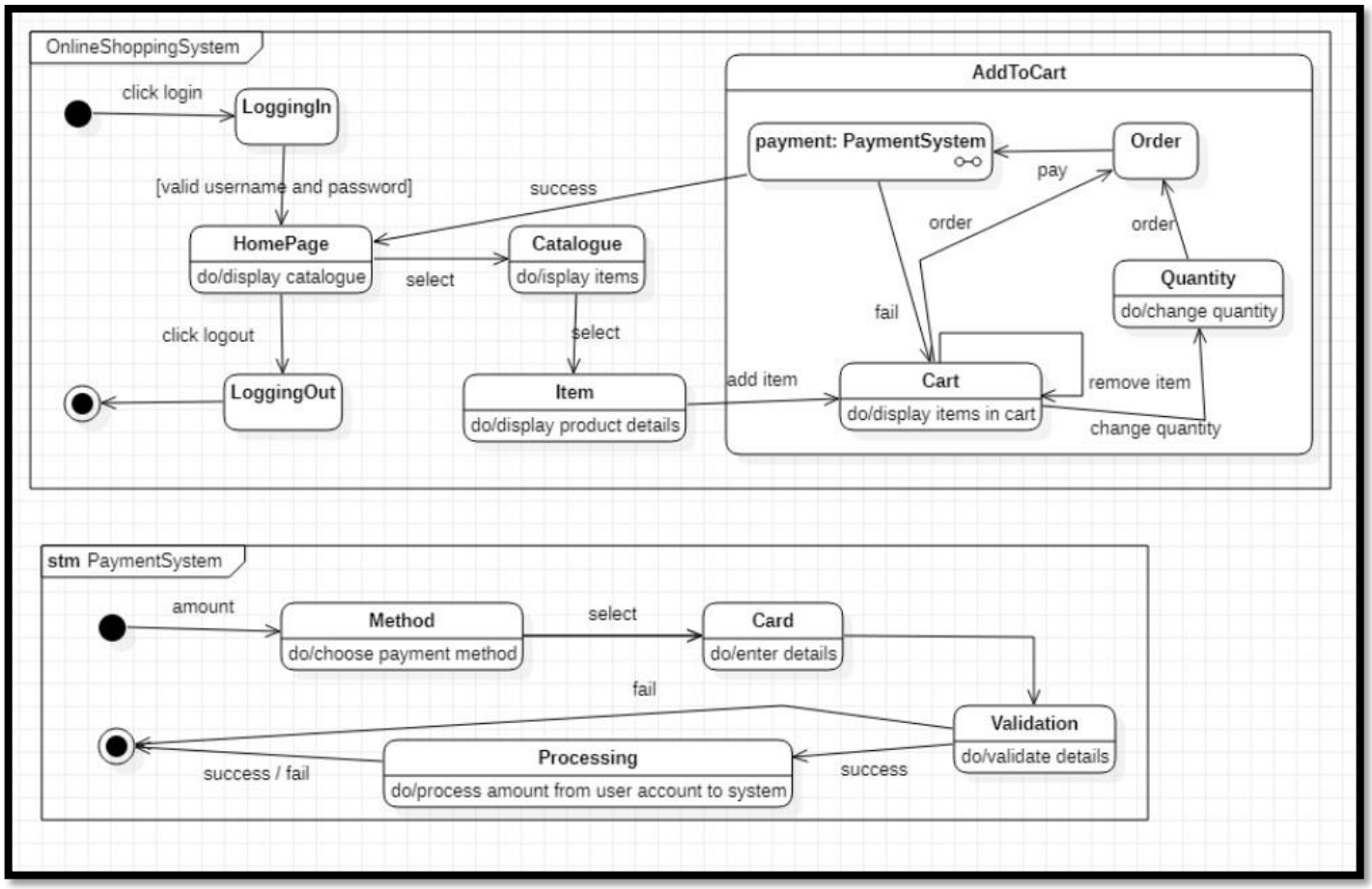
2. Advanced class diagram



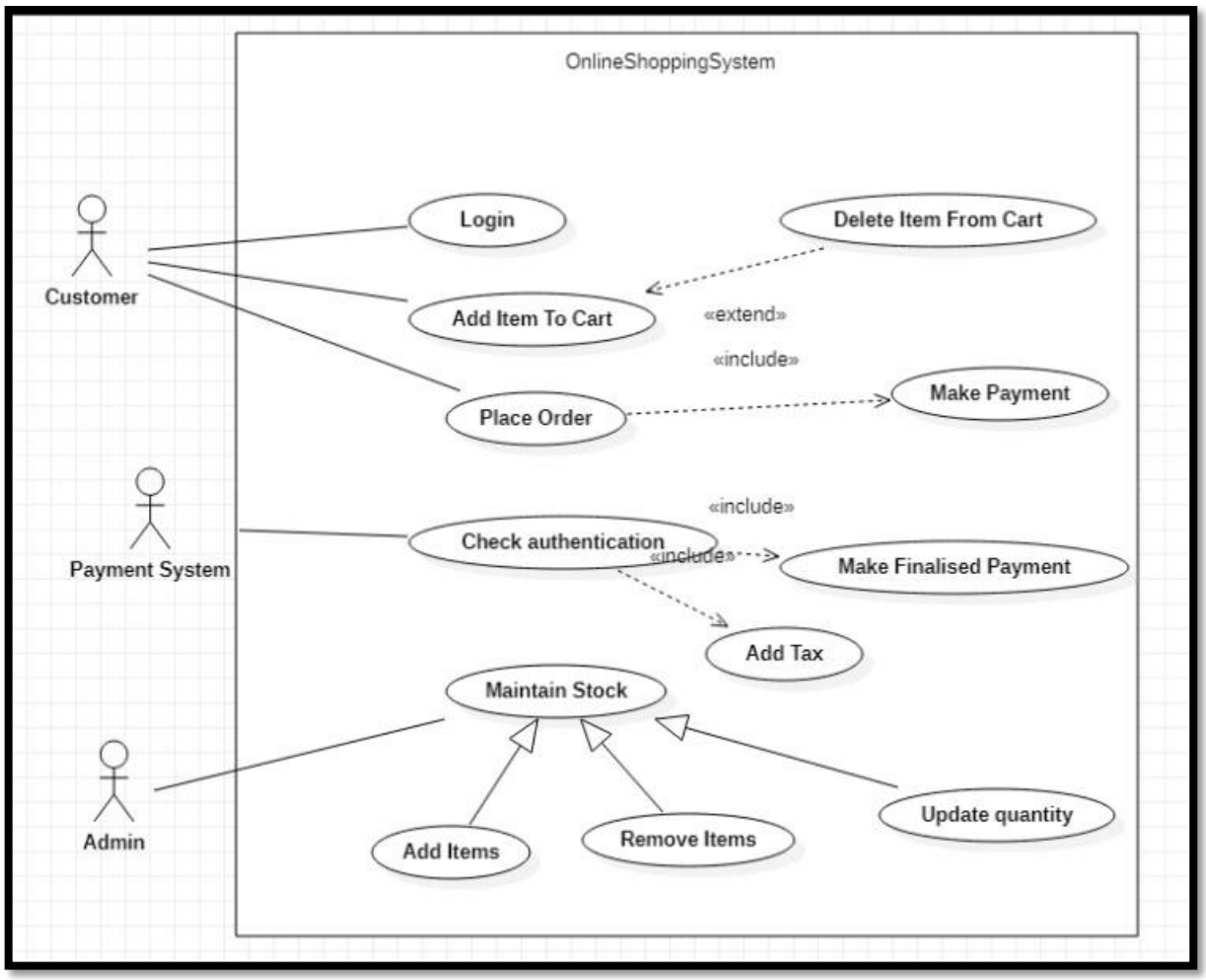

WebUser
+loginId: string
+password: string
+state: UserState
+getId()
+getPassword()
+getState()
+has
0..1
1
Customer
+name: string
+phone: int
+email: string
+age: int
+gender: string
+address: string
+edit()
+login()
+logout()
«enumeration»
UserState
new
active
blocked
banned
1
+owns
0..1
+has
1
1
ShoppingCart
+created: date
+addItem()
+deleteItem()
+has
1
1
Account
+billingAddress: string
+isClosed: boolean
+open: date
+closed: date
+search()
+edit()
«enumeration»
OrderStatus
orderPlaced
shipped
delivered
returned
1
*
+has
LineItem
+quantity: int
+price: int
1
+places
*
Product
+name: string
+supplier: string
+getDetails()
+readReview()
1..*
+consits
1
Order
+number: int
+ordered: date
+shipped: date
+shipFrom: string
+status: OrderStatus
+total: int
+checkStatus()
+cancelOrder()
+completes
1
1
Payment
+total: int
+details: string
+mode: string
+makePayment()
+cancelPayment()
NetBanking
+cardNumber: string
+paymentMade: date
COD
+noOfNotes: int

3. Advanced state diagram

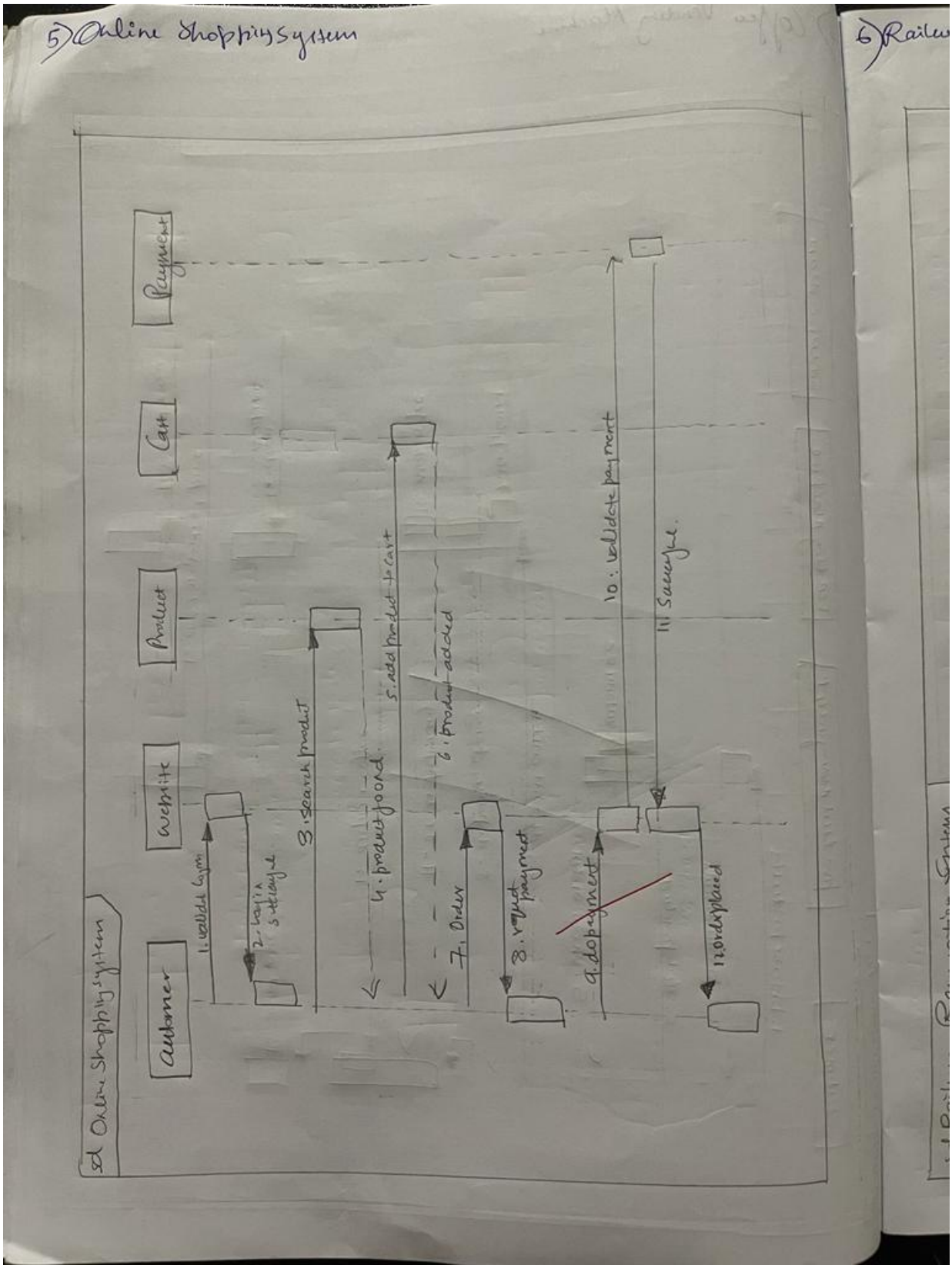


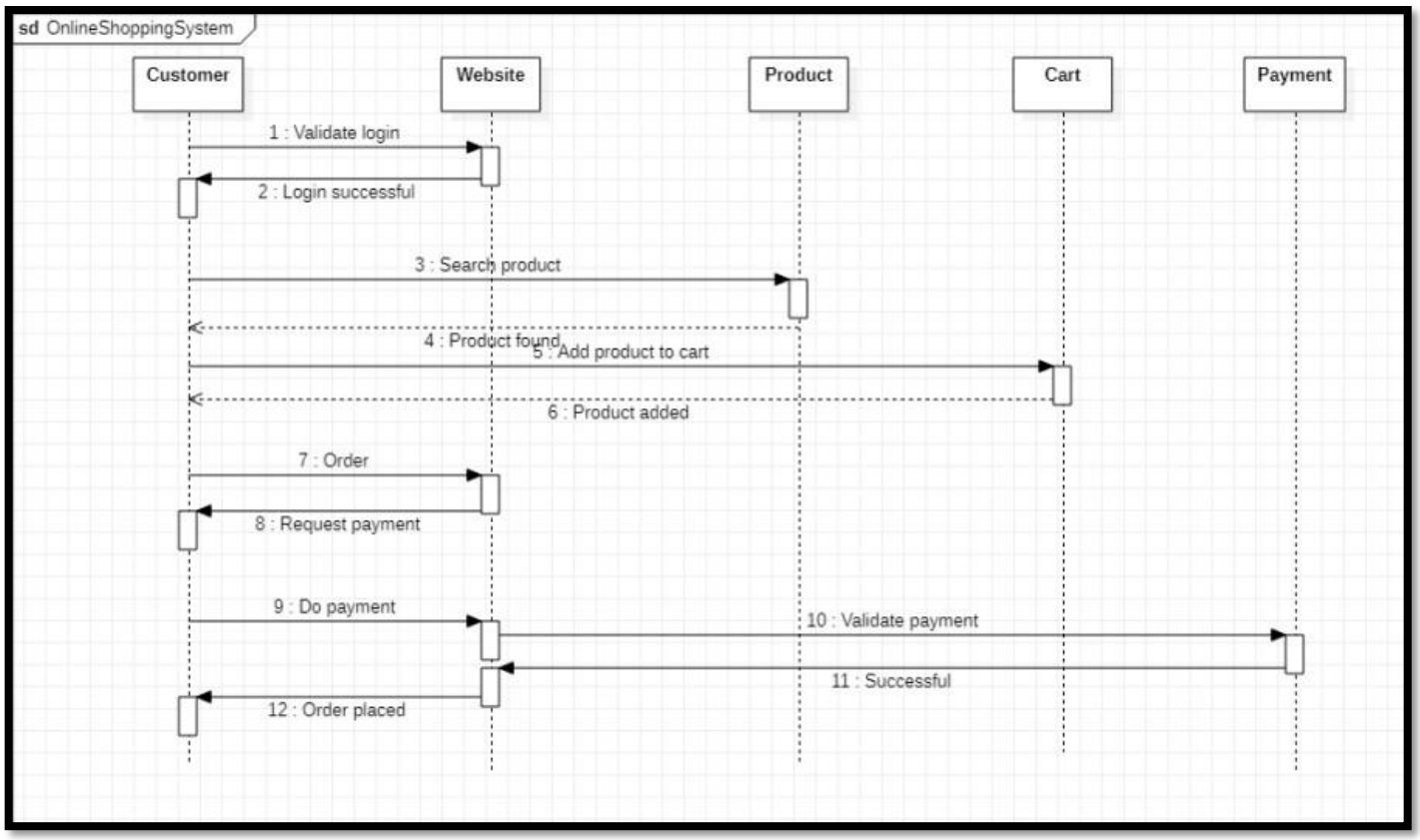

OnlineShoppingSystem
click login
LoggingIn
[valid username and password]
success
HomePage
do/display catalogue
select
Catalogue
do/isplay items
click logout
select
LoggingOut
Item
do/display product details
add item
AddToCart
payment: PaymentSystem
pay
Order
order
order
Quantity
do/change quantity
fail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

4. Advanced use case diagram

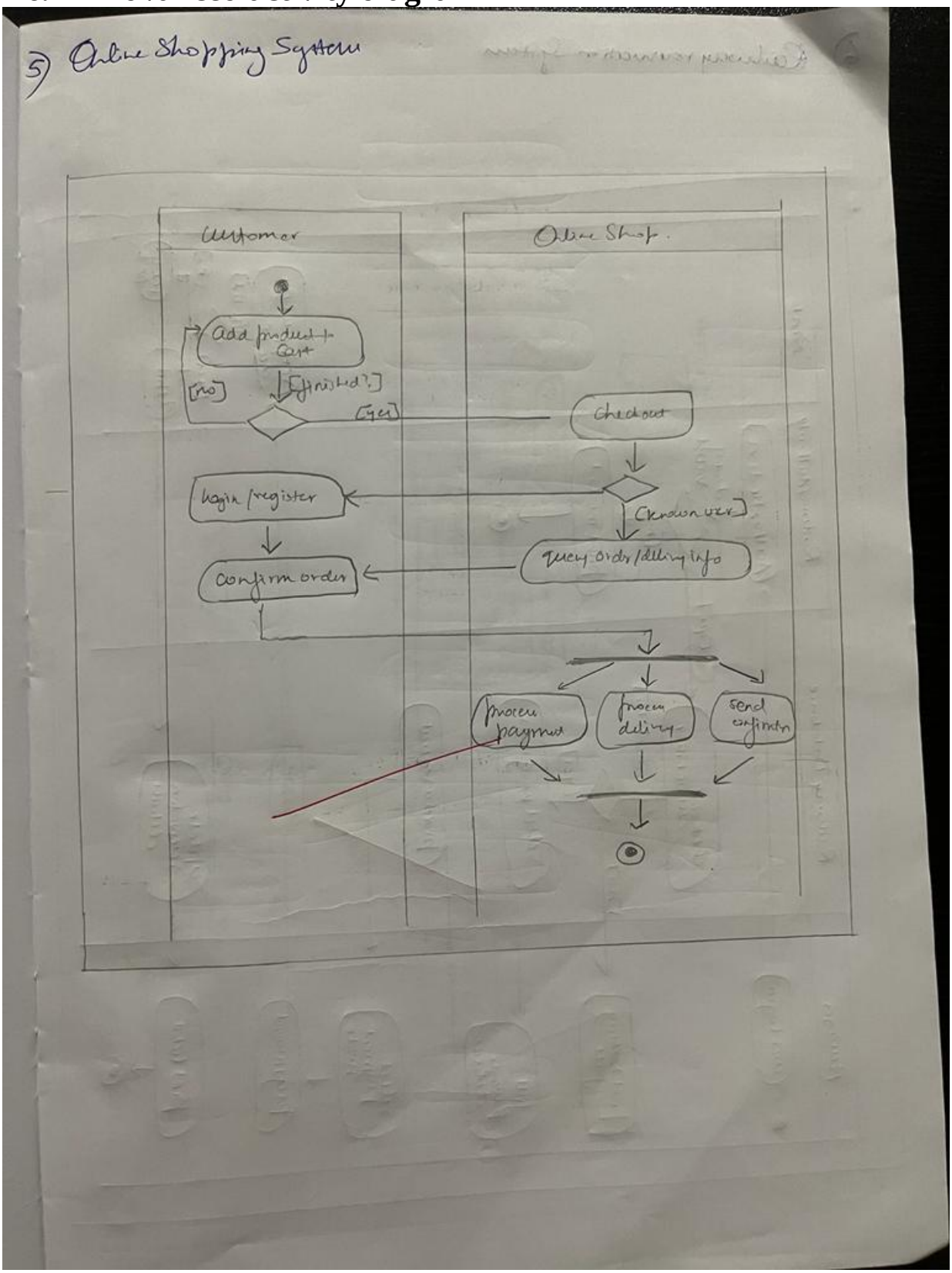

OnlineShoppingSystem
Customer
Login
Delete Item From Cart
Add Item To Cart
«extend»
«include»
Place Order
Make Payment
Payment System
«include»
Check authentication
«include»
Make Finalised Payment
Add Tax
Maintain Stock
Admin
Update quantity
Add Items
Remove Items

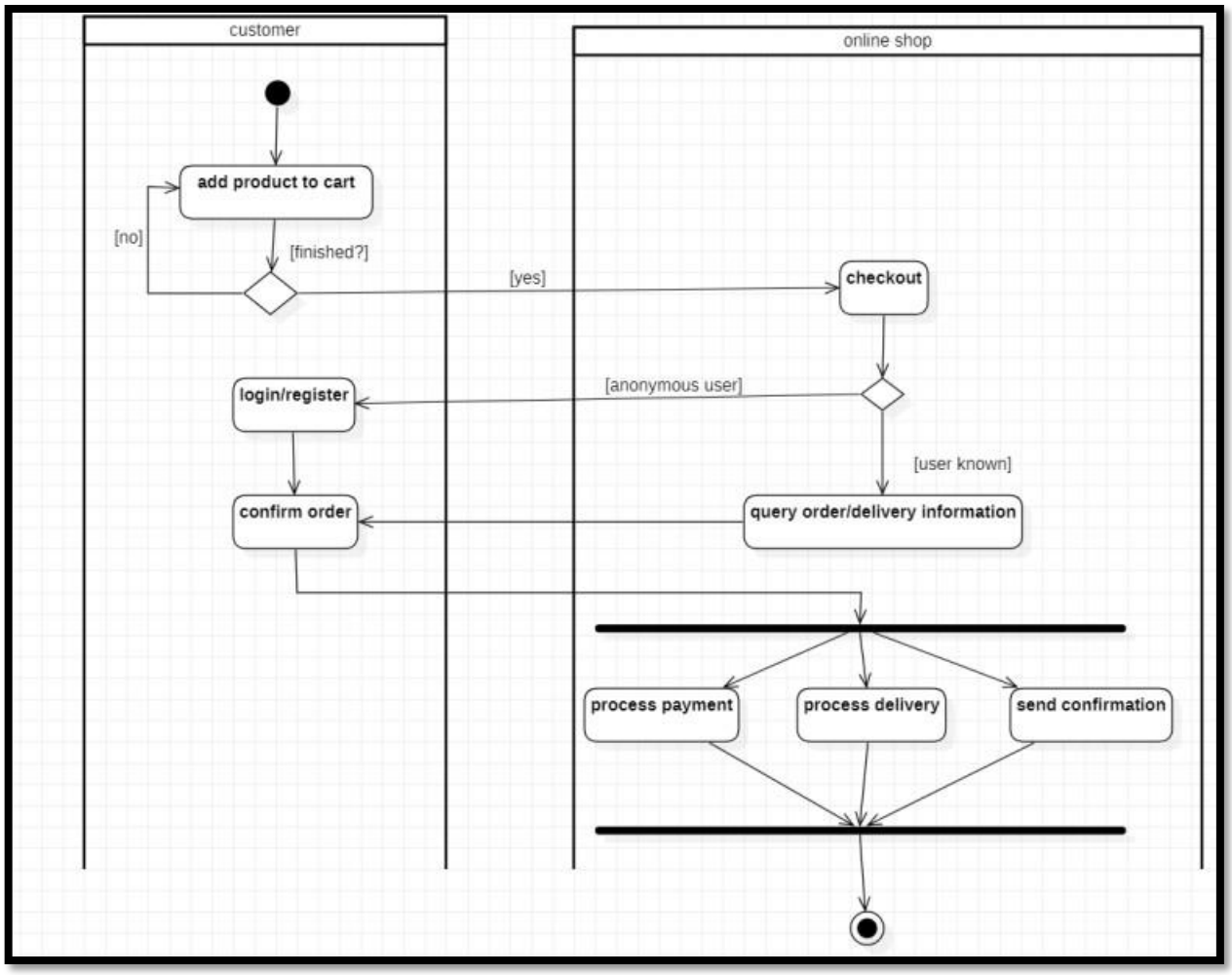
5. Advanced sequence diagram




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

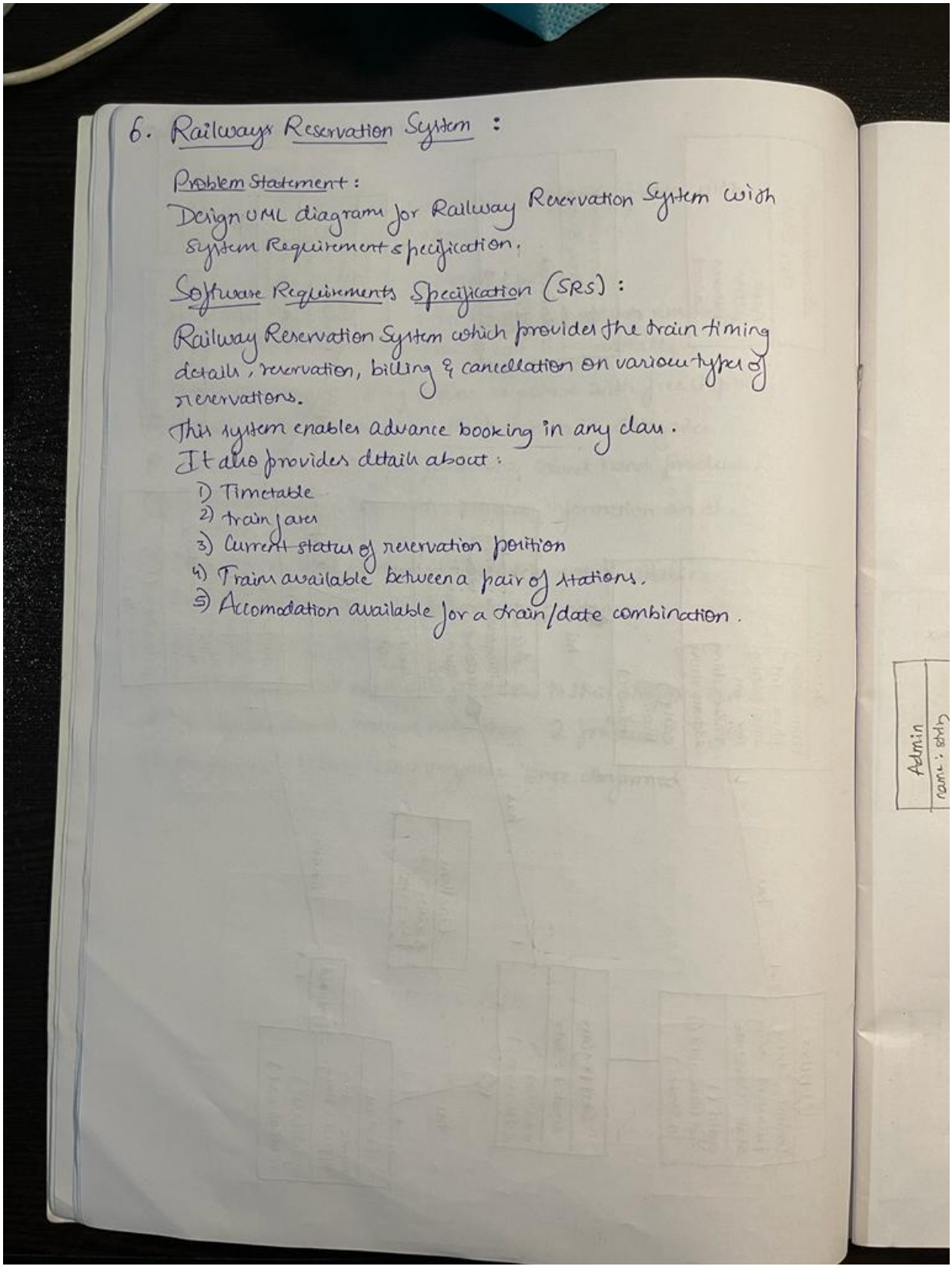
6. Advanced activity diagram




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

Exercise 6: Railway Reservation System

1. SRS



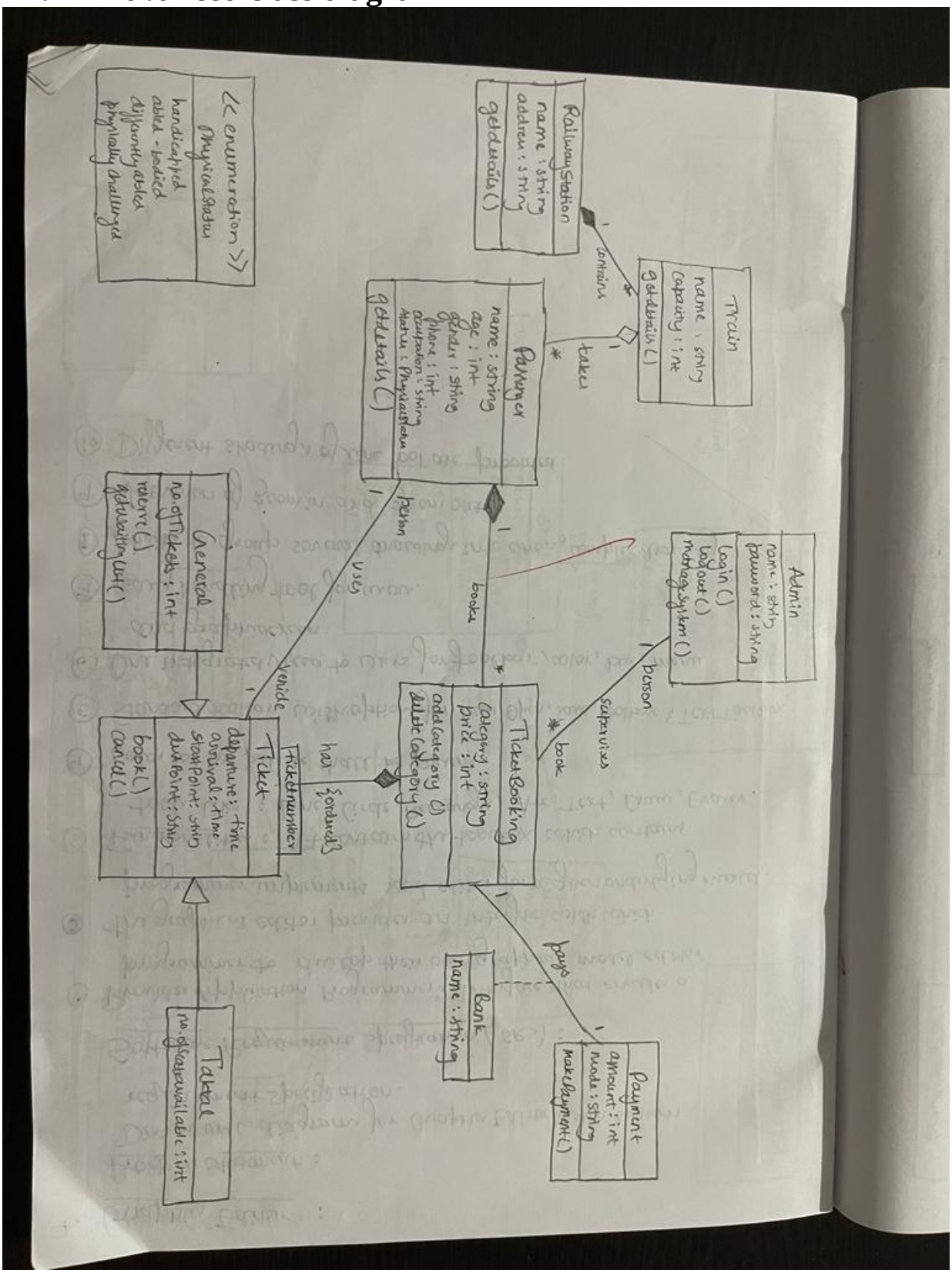
6. Railways Reservation System :

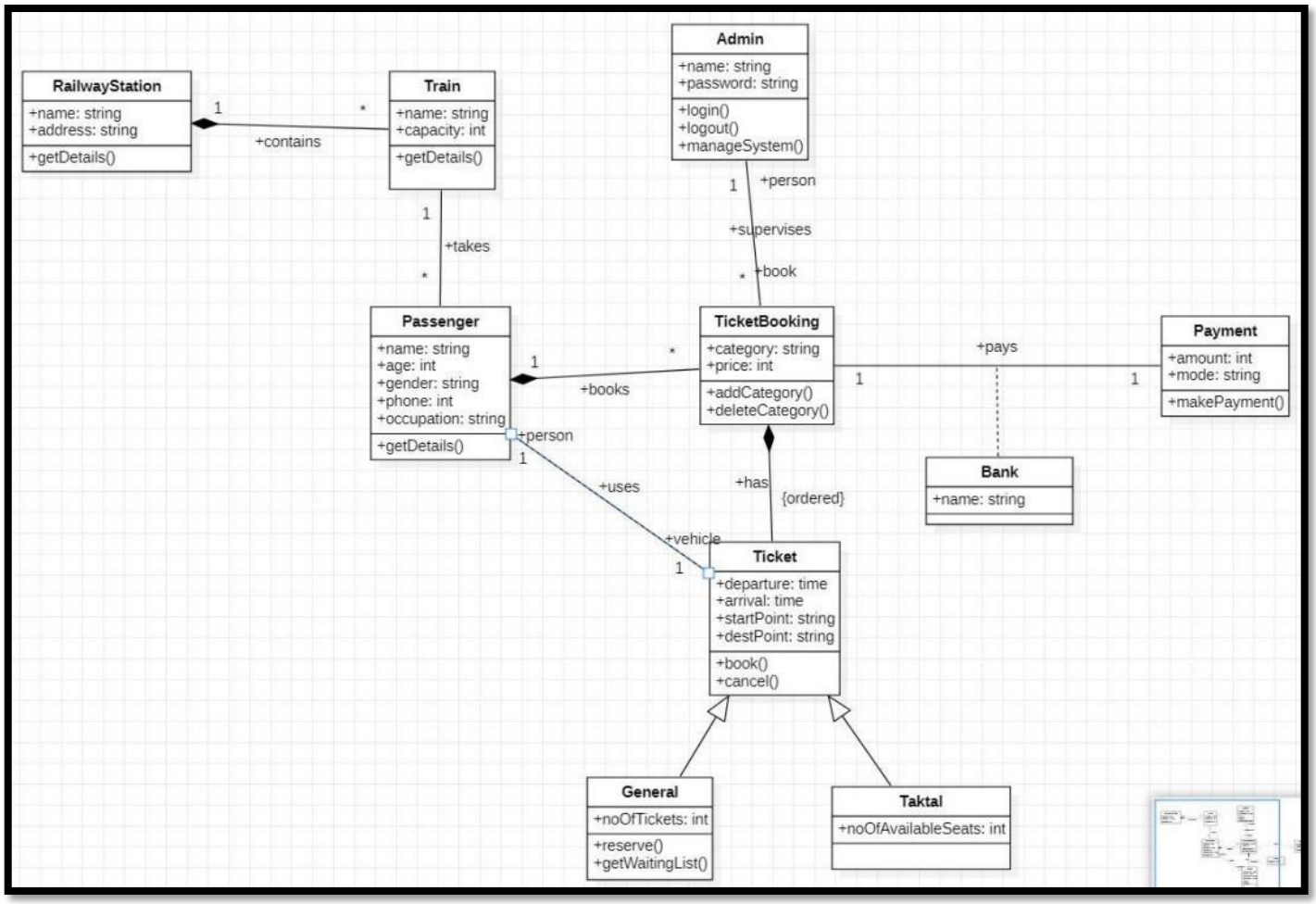
Problem Statement :
Design UML diagram for Railway Reservation System with System Requirement specification.

Software Requirements Specification (SRS) :
Railway Reservation System which provides the train timing details, reservation, billing & cancellation on various types of reservations.
This system enables advance booking in any class.
It also provides details about :
1) Timetable
2) Train fares
3) Current status of reservation position
4) Trains available between a pair of stations.
5) Accomodation available for a train/date combination.

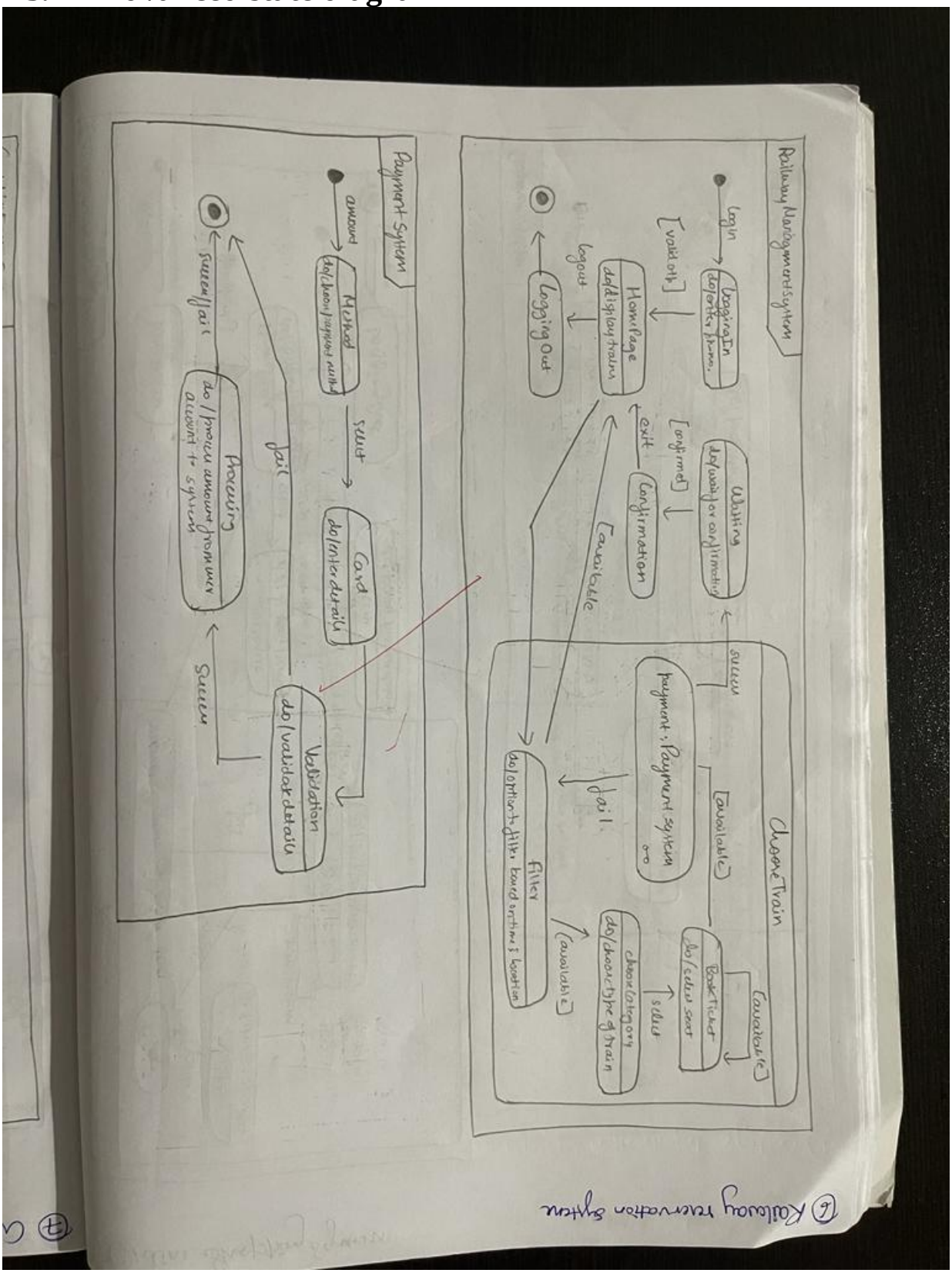
Admin
name : str

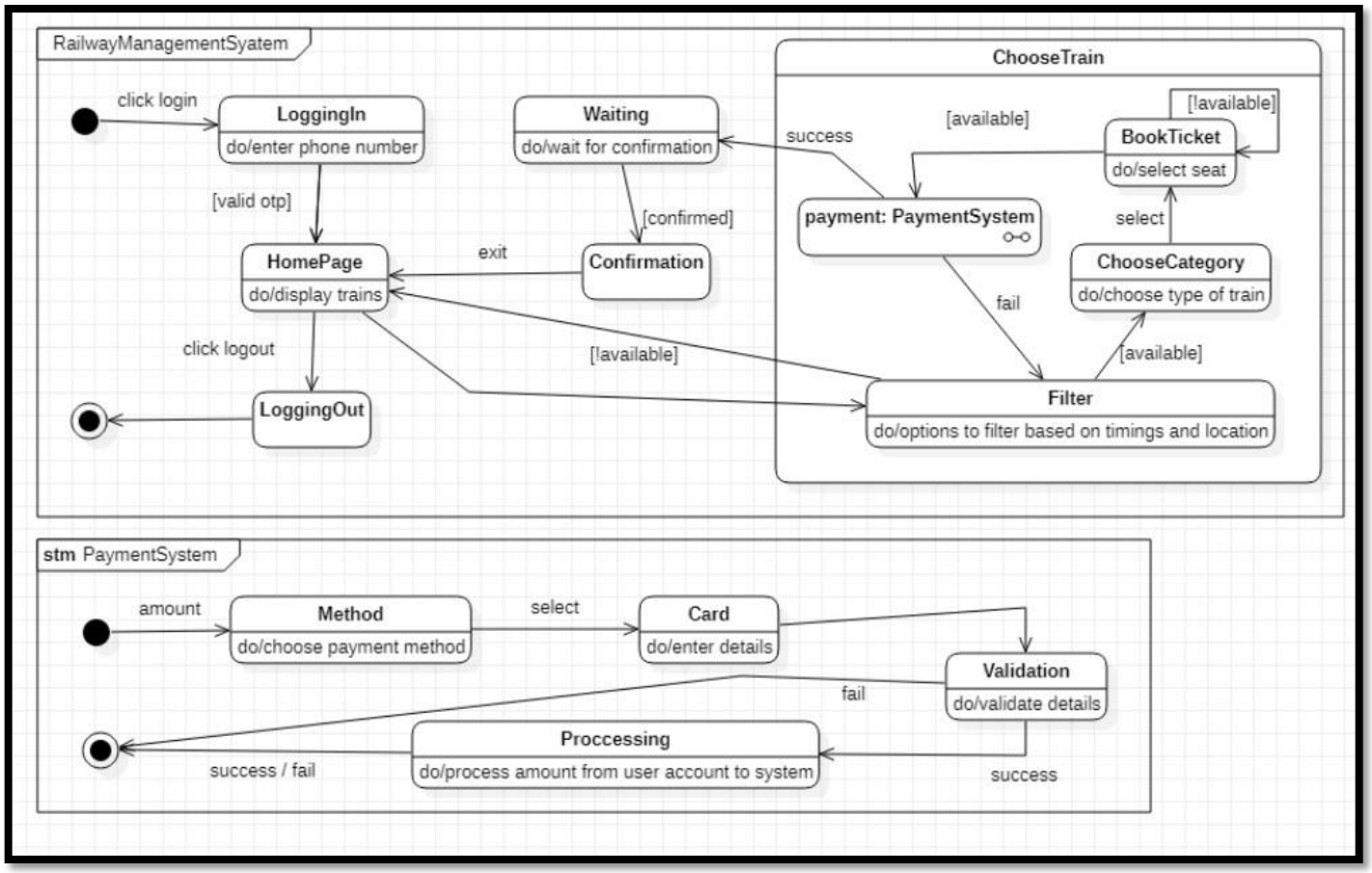
2. Advanced class diagram



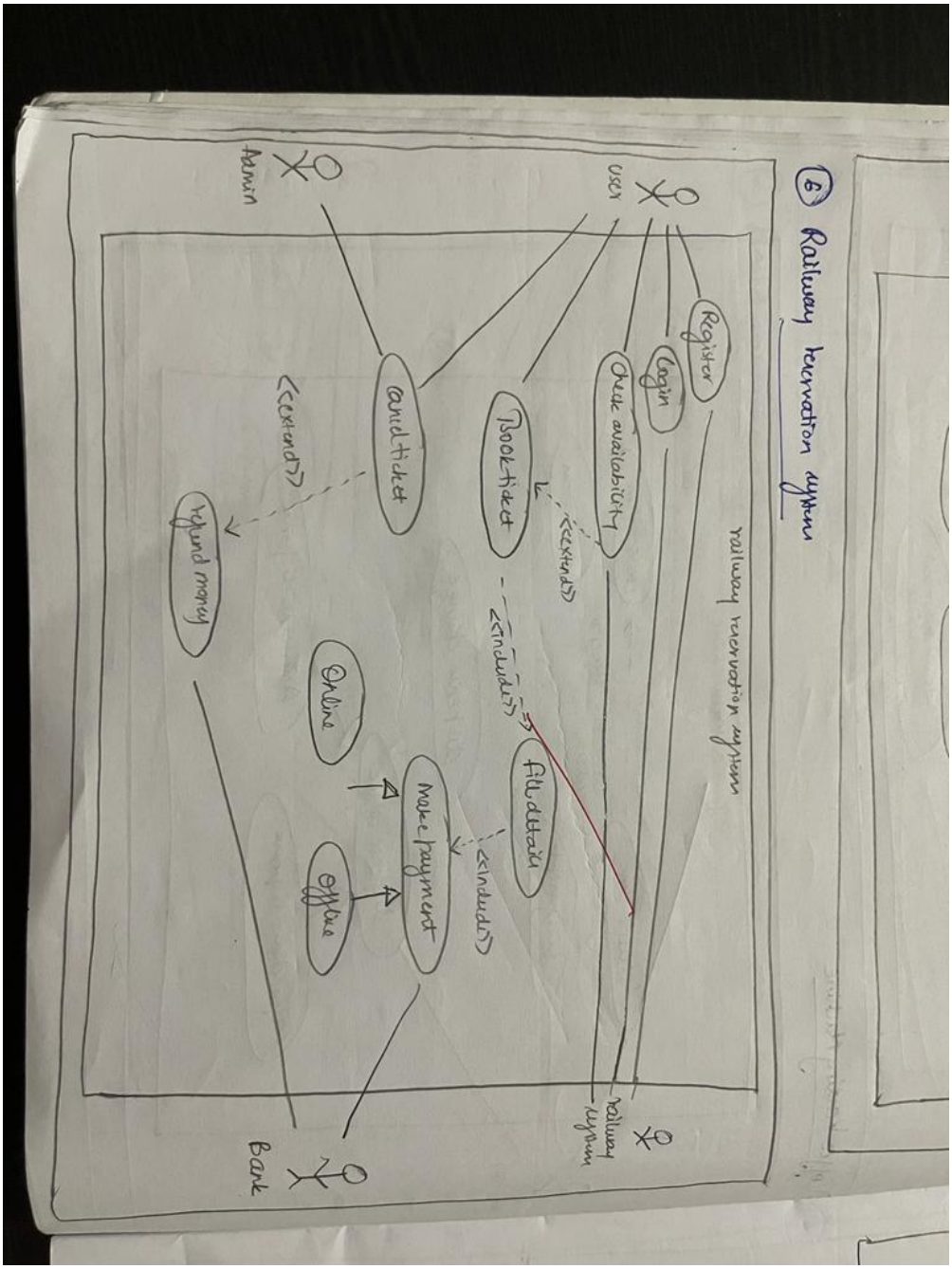

RailwayStation
+name: string
+address: string
+getDetails()
1
+contains
*
Train
+name: string
+capacity: int
+getDetails()
1
+takes
*
Admin
+name: string
+password: string
+login()
+logout()
+manageSystem()
1
+person
+supervises
*
+book
Passenger
+name: string
+age: int
+gender: string
+phone: int
+occupation: string
+getDetails()
1
+books
*
TicketBooking
+category: string
+price: int
+addCategory()
+deleteCategory()
1
+pays
1
Payment
+amount: int
+mode: string
+makePayment()
Bank
+name: string
+person
1
+uses
+has
{ordered}
+vehicle
1
Ticket
+departure: time
+arrival: time
+startPoint: string
+destPoint: string
+book()
+cancel()
General
+noOfTickets: int
+reserve()
+getWaitingList()
Taktal
+noOfAvailableSeats: int

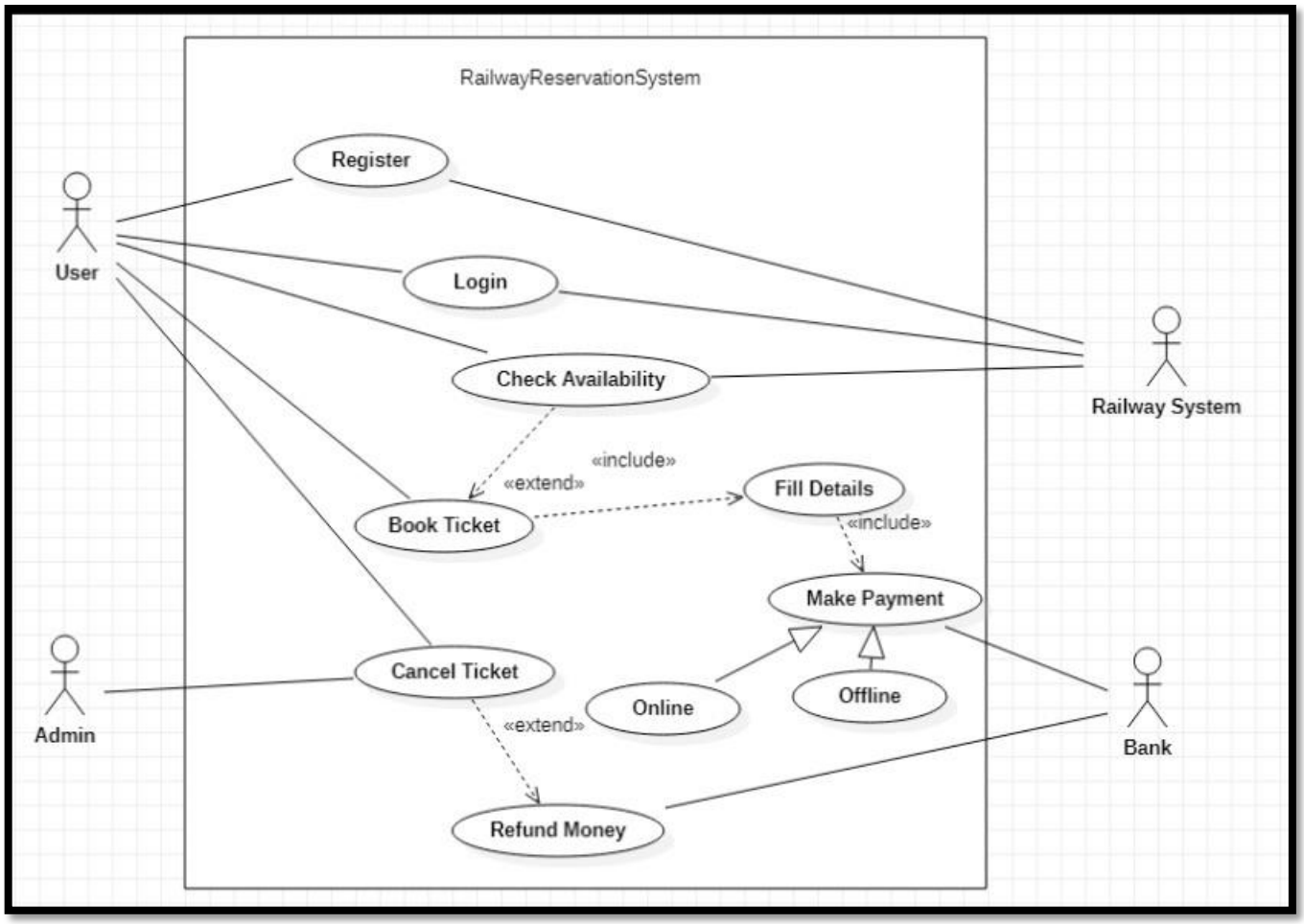
3. Advanced state diagram



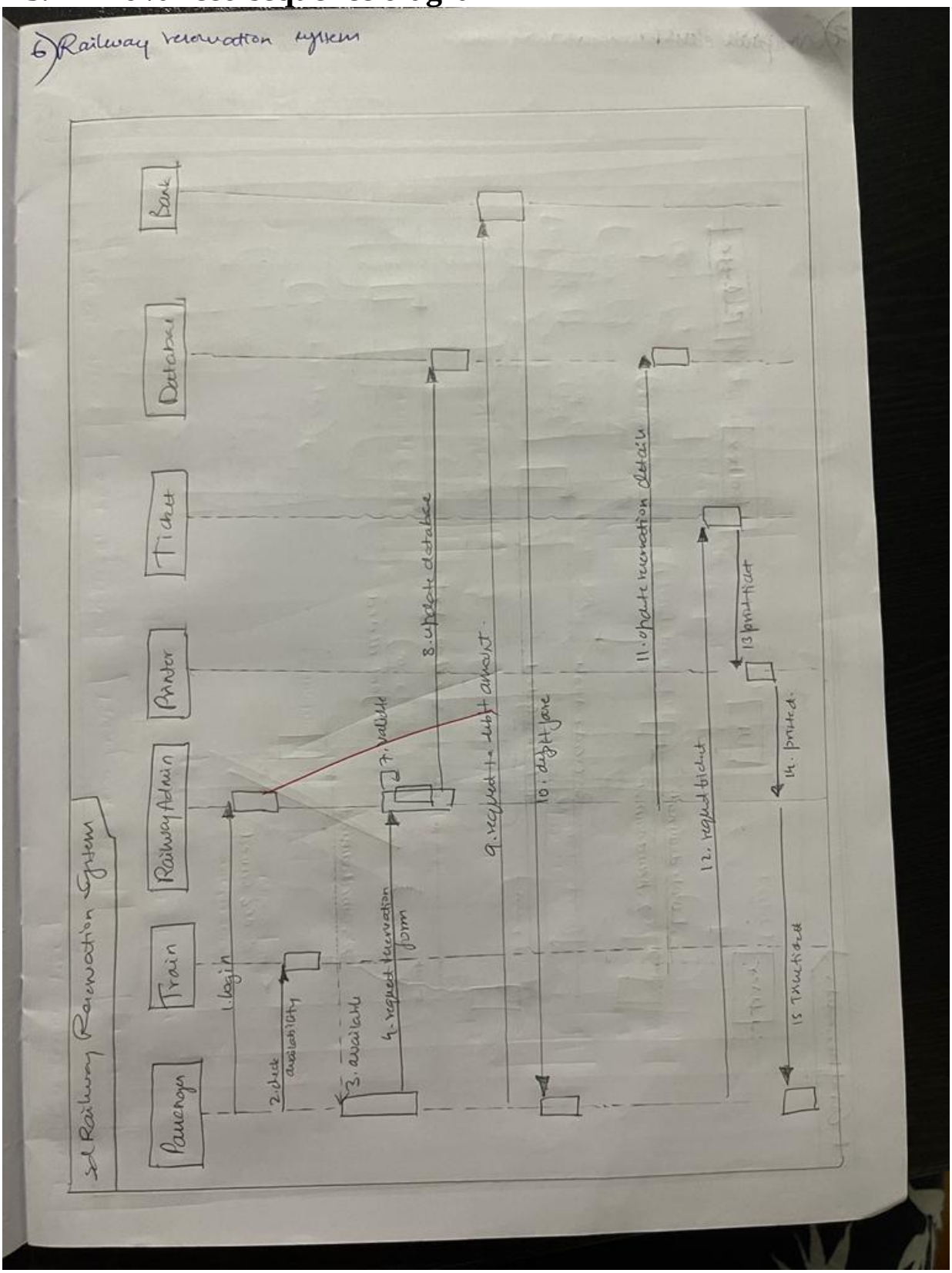

RailwayManagementSyatem
click login
LoggingIn
do/enter phone number
[valid otp]
HomePage
do/display trains
click logout
LoggingOut
exit
Confirmation
[confirmed]
Waiting
do/wait for confirmation
ChooseTrain
success
[available]
[!available]
BookTicket
do/select seat
payment: PaymentSystem
select
ChooseCategory
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

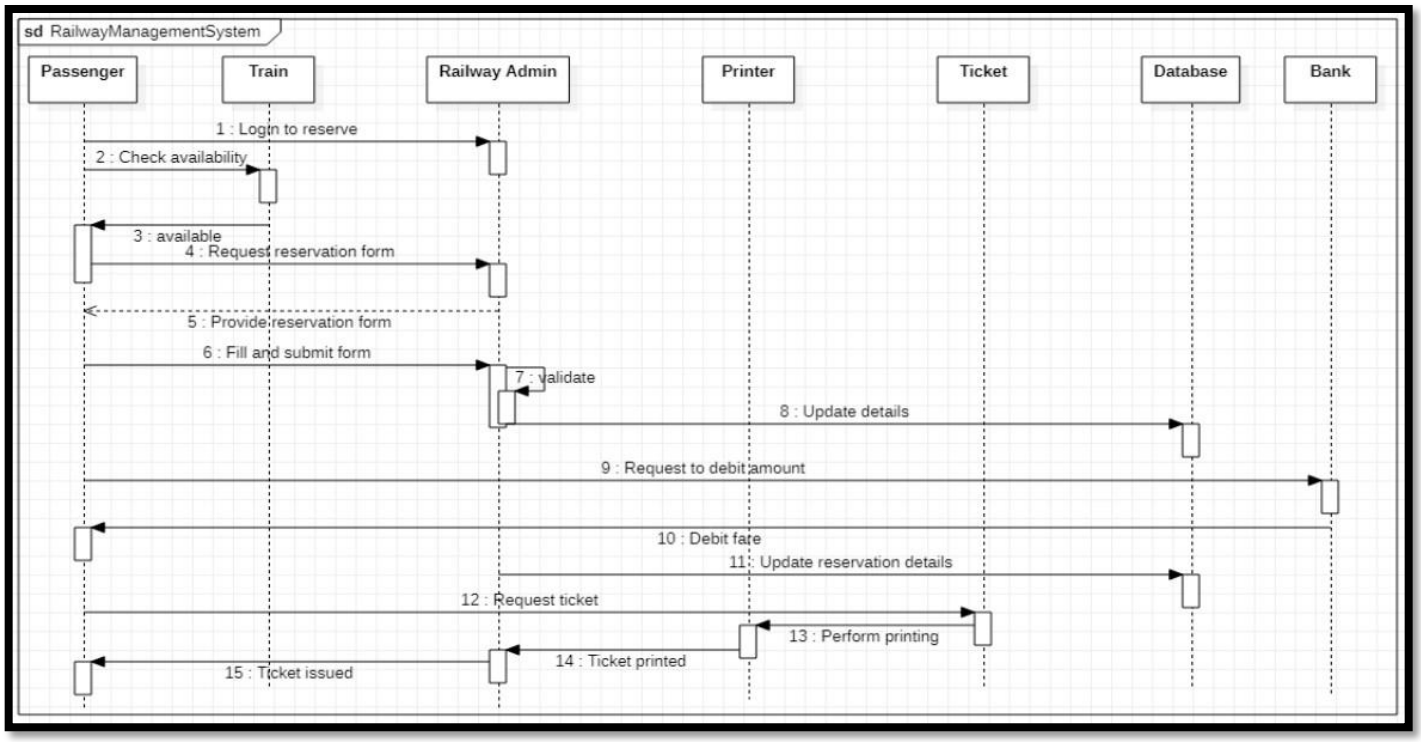
4. Advanced use case diagram



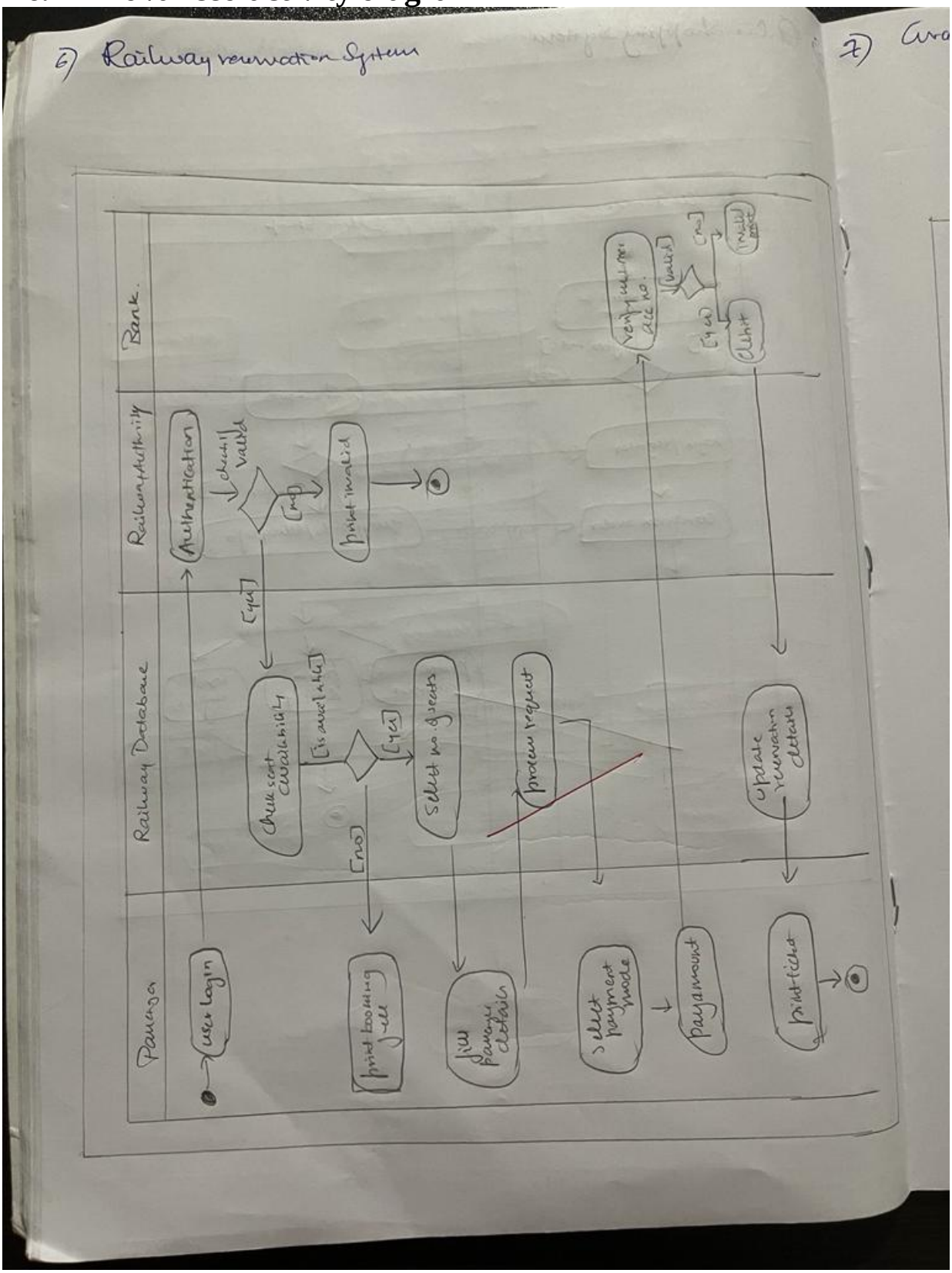

RailwayReservationSystem
Register
User
Login
Check Availability
Railway System
«include»
«extend»
Fill Details
Book Ticket
«include»
Make Payment
Admin
Cancel Ticket
Online
Offline
Bank
«extend»
Refund Money

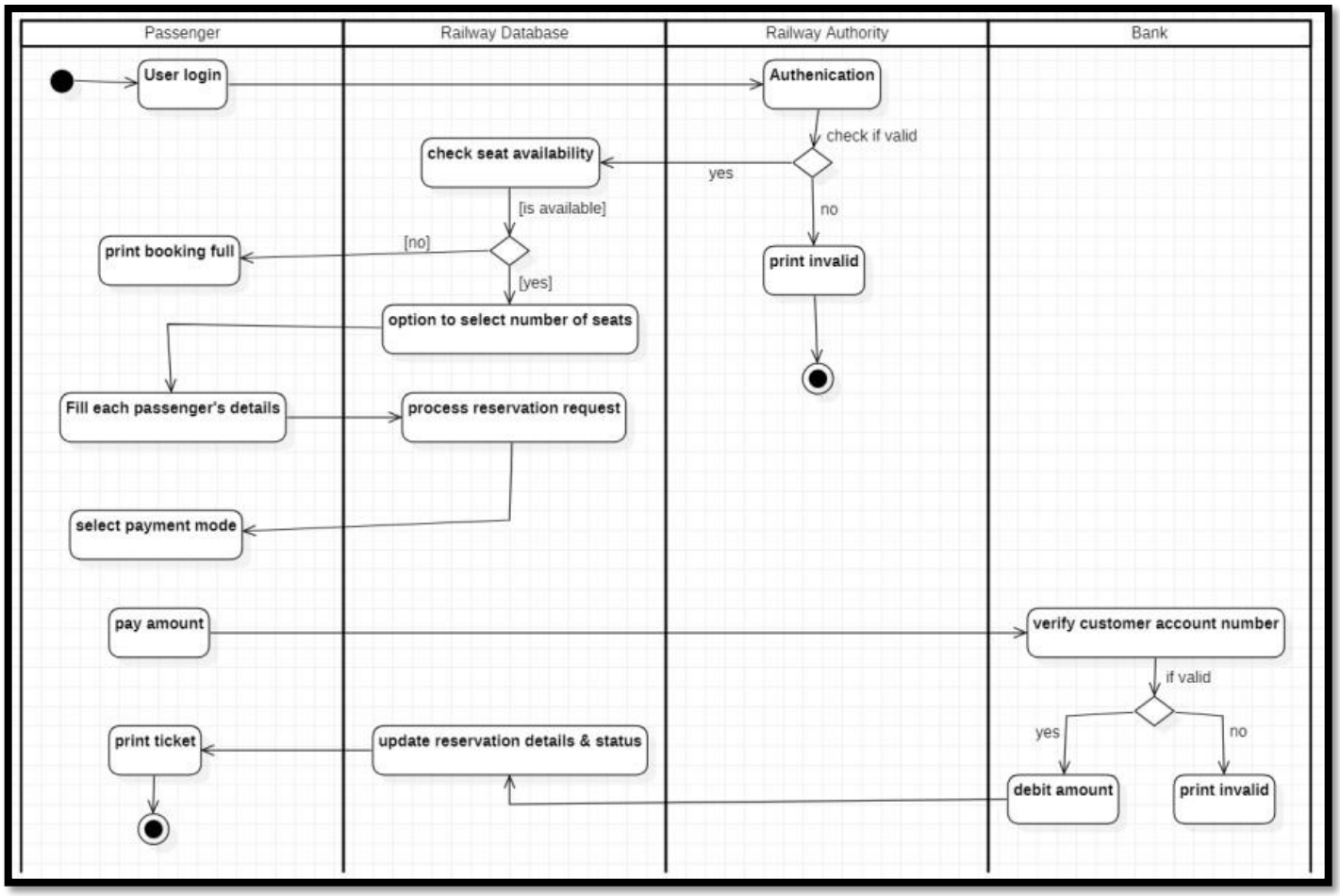
5. Advanced sequence diagram




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

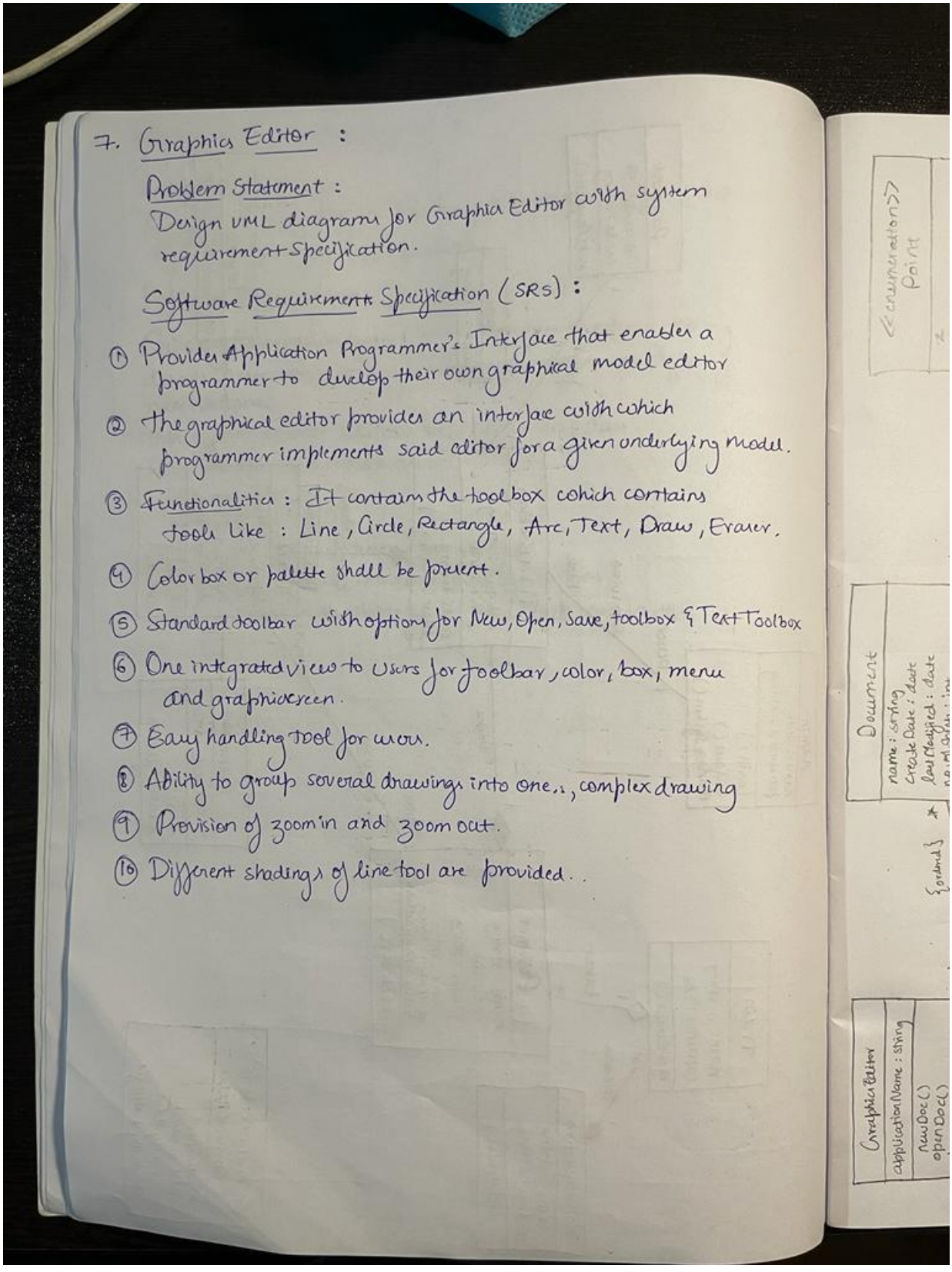
6. Advanced activity diagram




Passenger
Railway Database
Railway Authority
Bank
User login
Authenication
check if valid
check seat availability
yes
[is available]
no
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

Exercise 7: Graphics Editor System

1. SRS



7. Graphics Editor :

Problem Statement :
Design UML diagrams for Graphics Editor with system requirement Specification.

Software Requirement Specification (SRS) :

① Provides Application Programmer's Interface that enables a programmer to develop their own graphical model editor

② the graphical editor provides an interface with which programmer implements said editor for a given underlying model.

③ Functionalities : It contains the toolbox which contains tools like : Line, Circle, Rectangle, Arc, Text, Draw, Eraser.

④ Color box or palette shall be present.

⑤ Standard toolbar with options for New, Open, Save, toolbox & Text Toolbox

⑥ One integrated views to users for toolbar, color, box, menu and graphic screen.

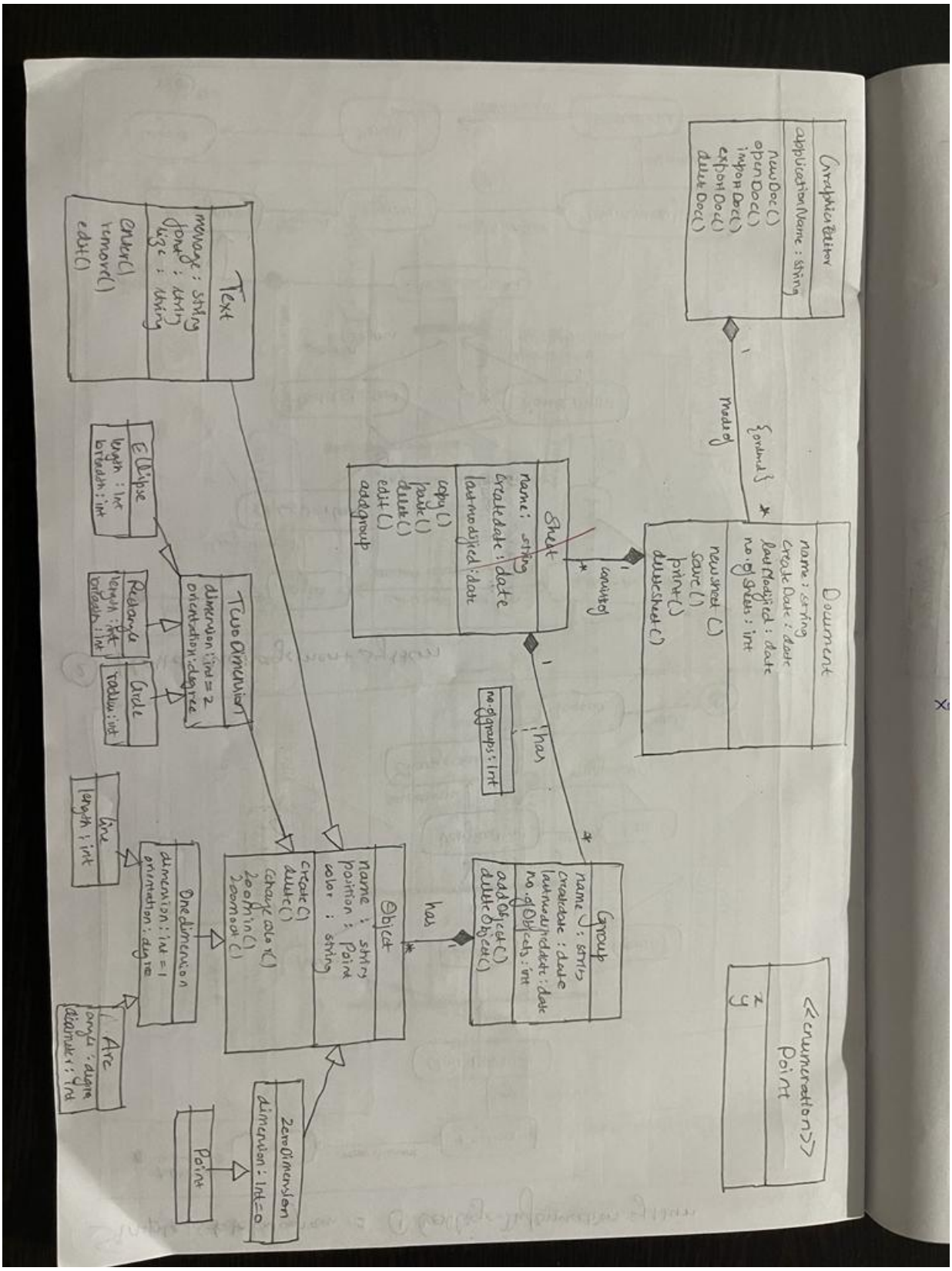
⑦ Easy handling tool for users.

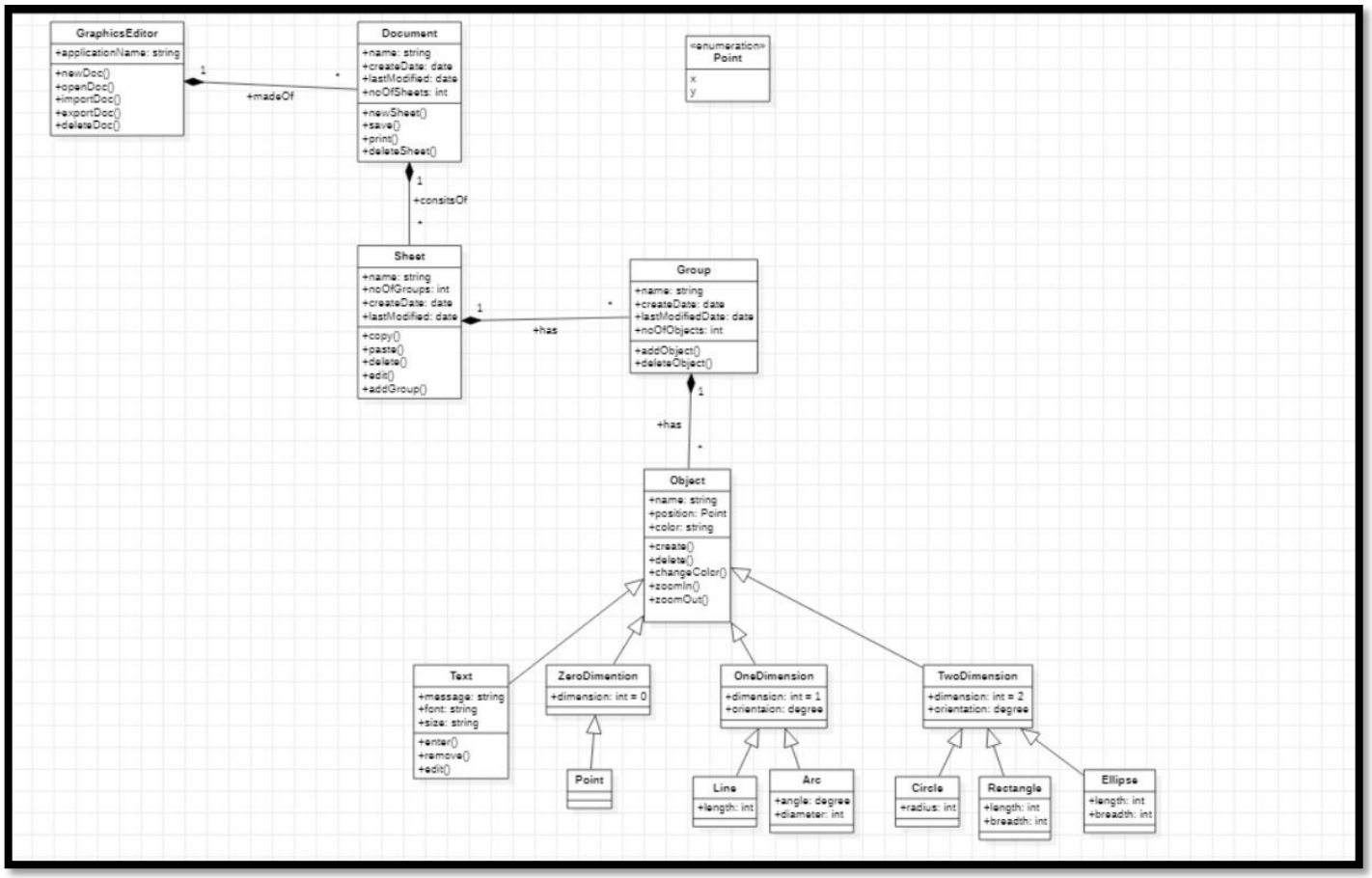
⑧ Ability to group several drawings into one., complex drawing

⑨ Provision of zoom in and zoom out.

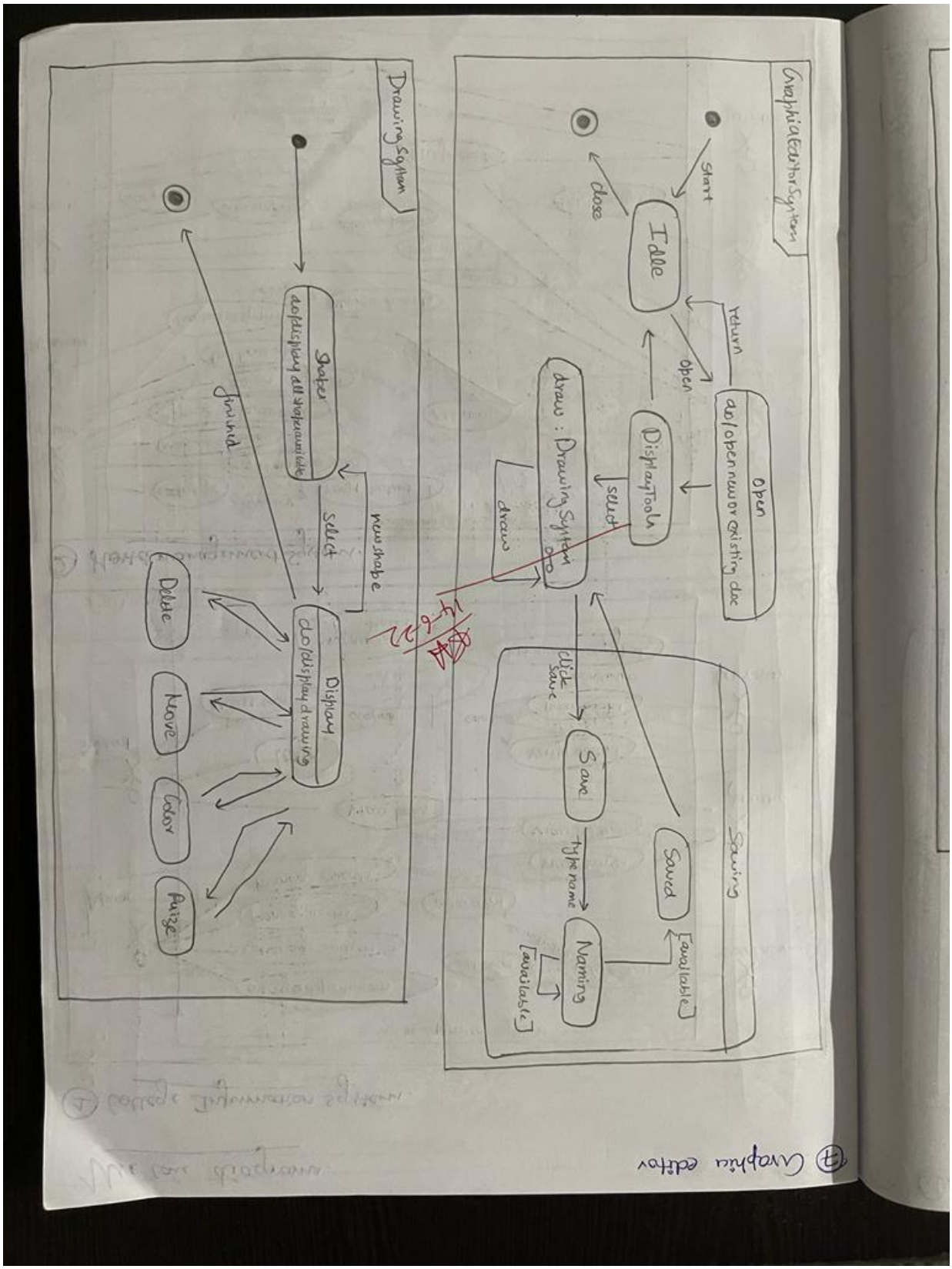
⑩ Different shadings of line tool are provided.

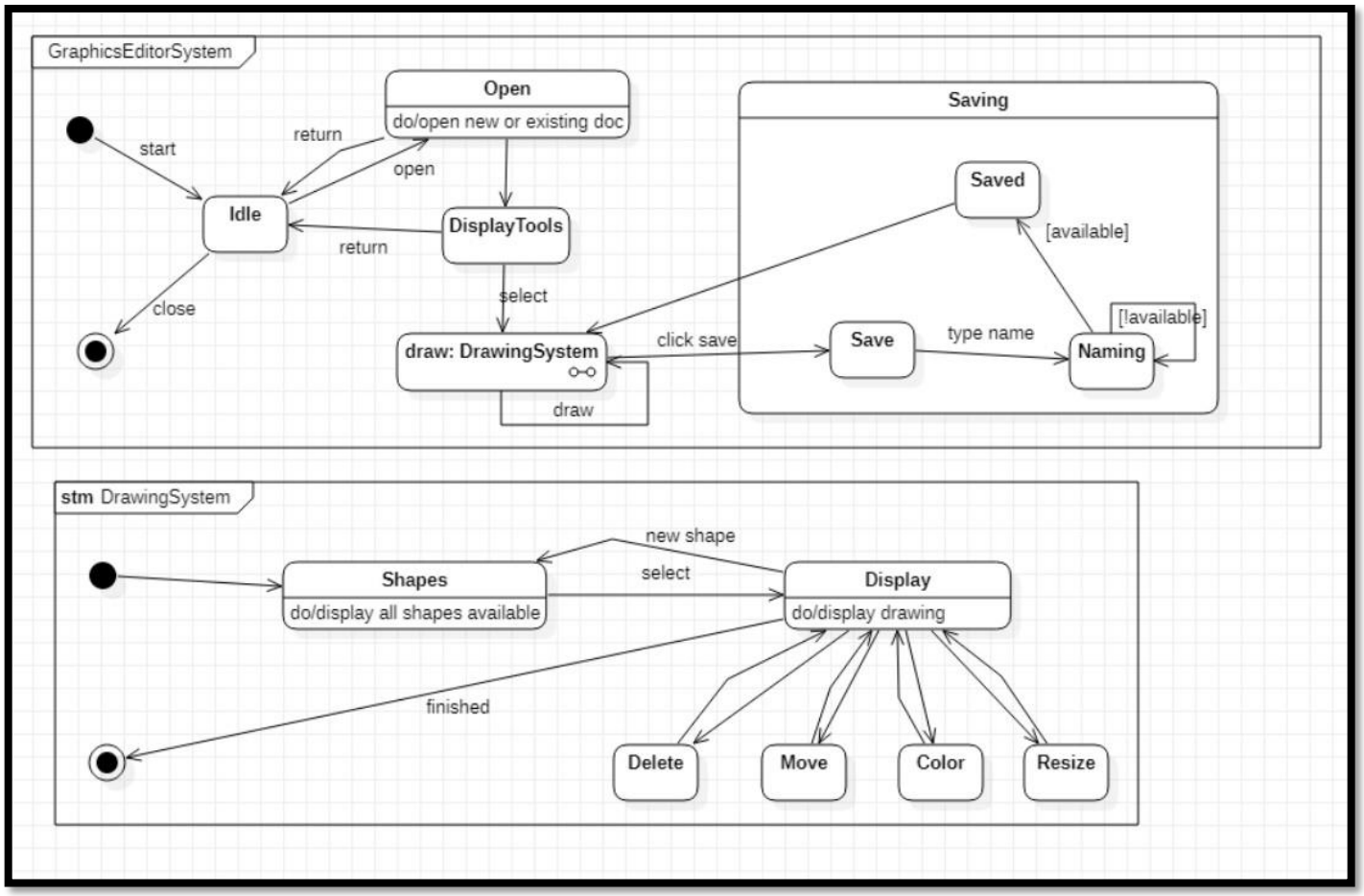
2. Advanced class diagram



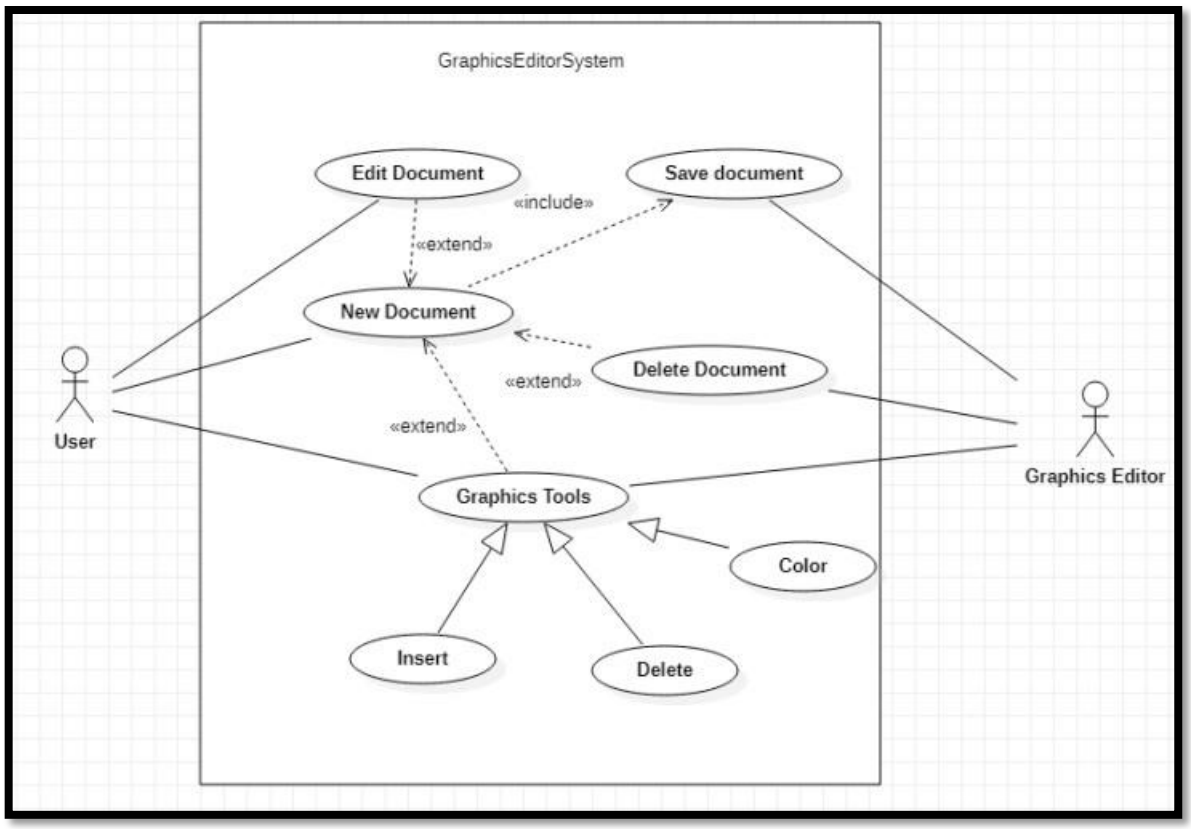

GraphicsEditor
+applicationName: string
+newDoc()
+openDoc()
+importDoc()
+exportDoc()
+deleteDoc()
1
+madeOf
*
Document
+name: string
+createDate: date
+lastModified: date
+noOfSheets: int
+newSheet()
+save()
+print()
+deleteSheet()
«enumeration»
Point
x
y
1
+consitsOf
*
Sheet
+name: string
+noOfGroups: int
+createDate: date
+lastModified: date
+copy()
+paste()
+delete()
+edit()
+addGroup()
1
+has
*
Group
+name: string
+createData: date
+lastModifiedDate: date
+noOfObjects: int
+addObject()
+deleteObject()
1
+has
*
Object
+name: string
+position: Point
+color: string
+create()
+delete()
+changeColor()
+zoomIn()
+zoomOut()
Text
+message: string
+font: string
+size: string
+enter()
+remove()
+edit()
ZeroDimention
+dimension: int = 0
Point
OneDimension
+dimension: int = 1
+orientaion: degree
Line
+length: int
Arc
+angle: degree
+diameter: int
TwoDimension
+dimension: int = 2
+orientation: degree
Circle
+radius: int
Rectangle
+length: int
+breadth: int
Ellipse
+length: int
+breadth: int

3. Advanced state diagram

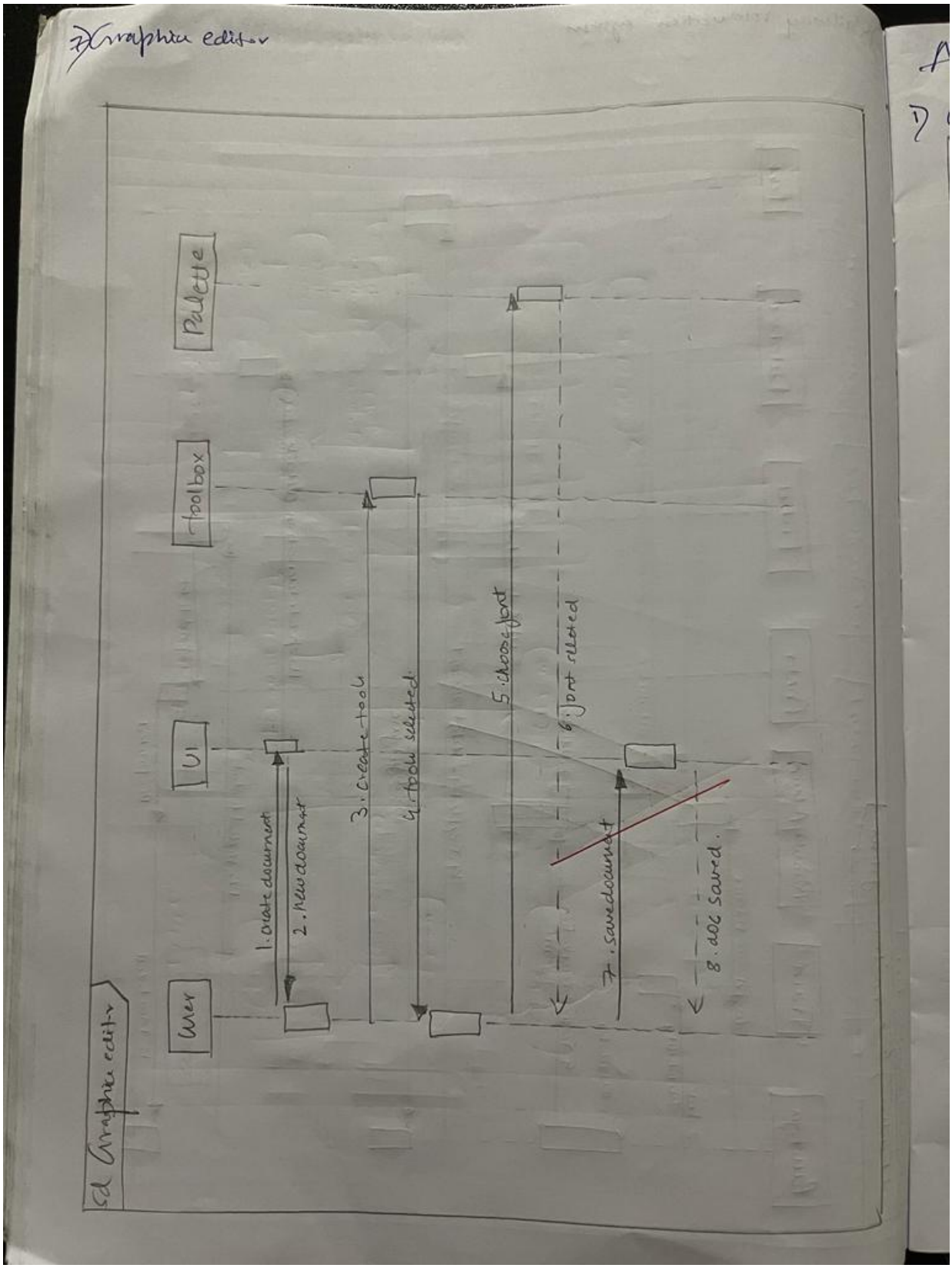


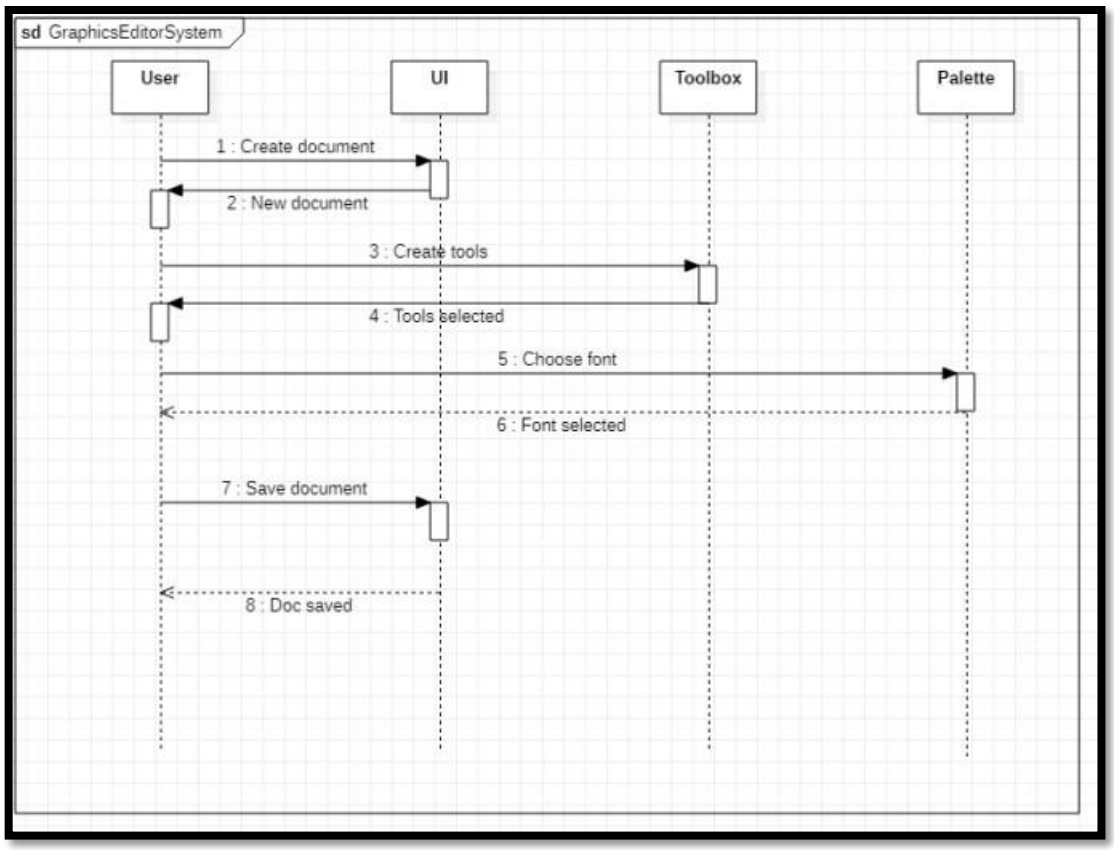

GraphicsEditorSystem
Open
do/open new or existing doc
start
return
open
Idle
DisplayTools
return
close
select
draw: DrawingSystem
draw
Saving
Saved
[available]
click save
Save
type name
Naming
[!available]
stm DrawingSystem
new shape
select
Shapes
do/display all shapes available
Display
do/display drawing
finished
Delete
Move
Color
Resize

4. Advanced use case diagram

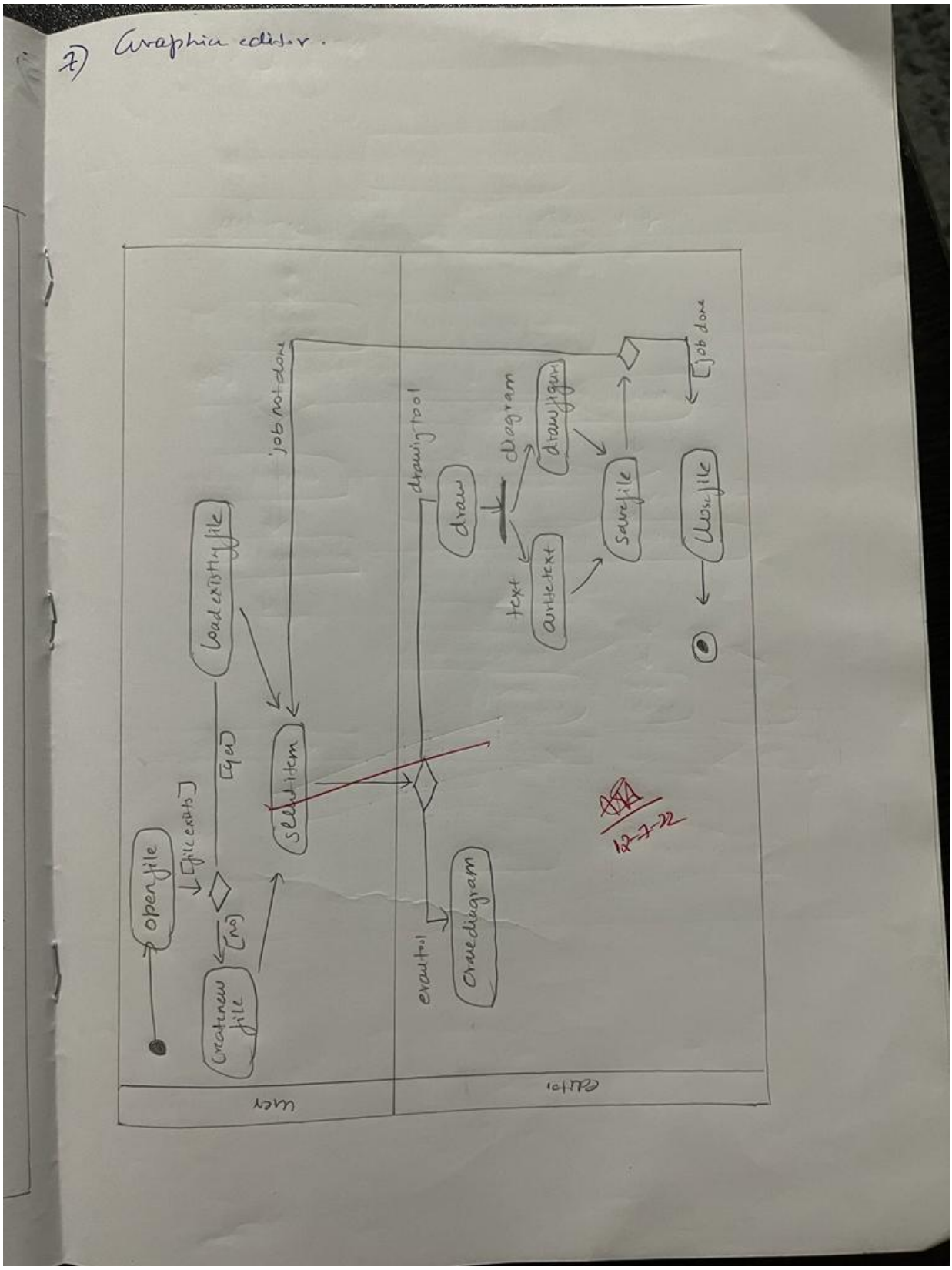

GraphicsEditorSystem
Edit Document
Save document
«include»
«extend»
New Document
Delete Document
«extend»
«extend»
User
Graphics Editor
Graphics Tools
Color
Insert
Delete

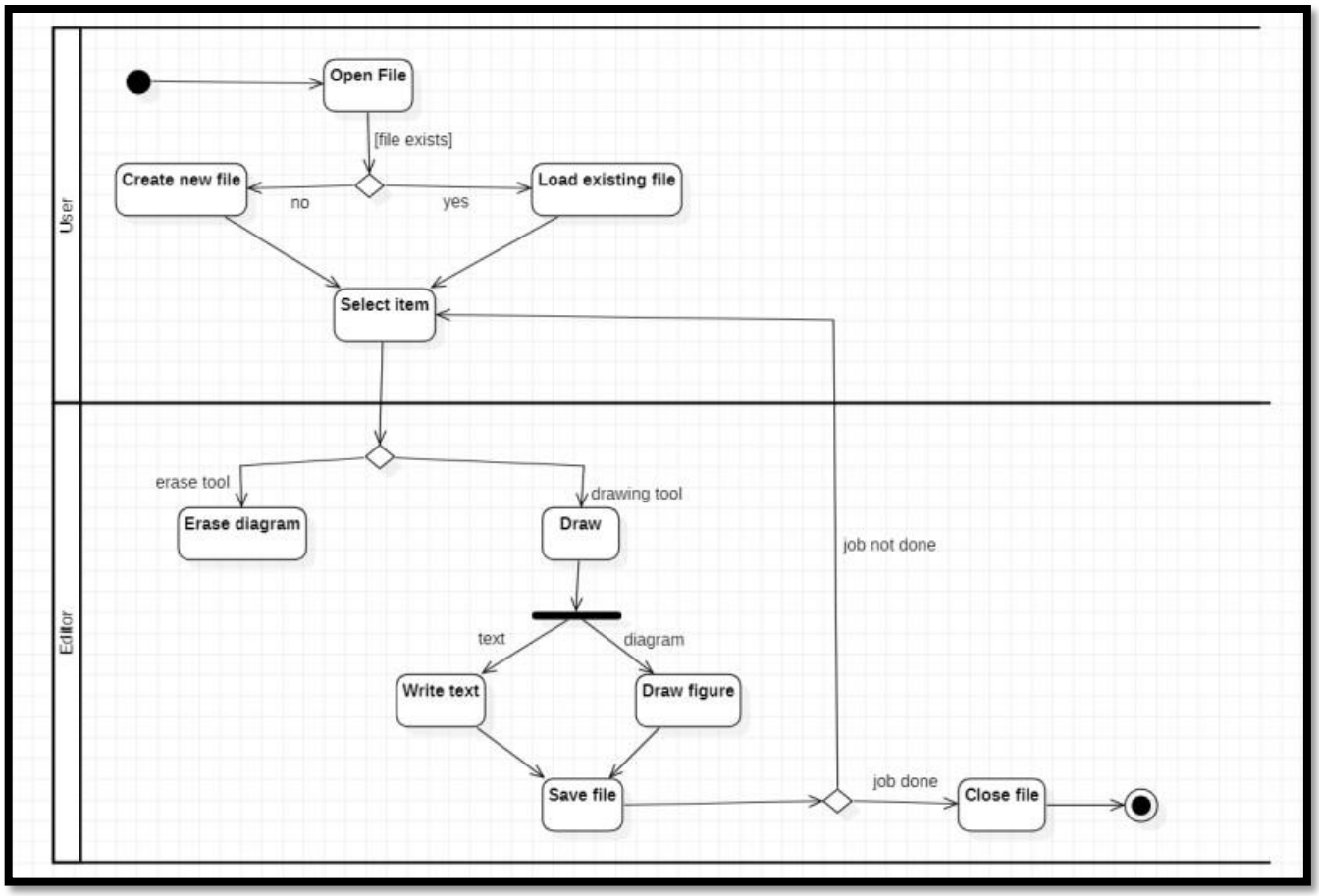
5. Advanced sequence diagram




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

6. Advanced activity diagram




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