VISVESVARAYA TECHNOLOGICAL UNIVERSITY

"JnanaSangama", Belgaum -590014, Karnataka.

LAB REPORT
on

OBJECT ORIENTED MODELLING AND DESIGN

Submitted by

PRATHIKSHA KAMATH(1BM19CS118)

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
May-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 MODELLING AND DESIGN**" carried out by **PRATHIKSHA KAMATH(1BM19CS118),** 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 year 2022. The Lab report has been approved as it satisfies the academic requirements in respect of an Object **Oriented Modelling and Design - (20CS6PCOMD)** work prescribed for the said degree.

Name of the Lab-Incharge
Designation
Department of CSE
BMSCE, Bengaluru

Dr. Nandini Vineeth
Assistant Professor
Department of CSE
BMSCE, Bengaluru

`

Index Sheet

Course Outcome

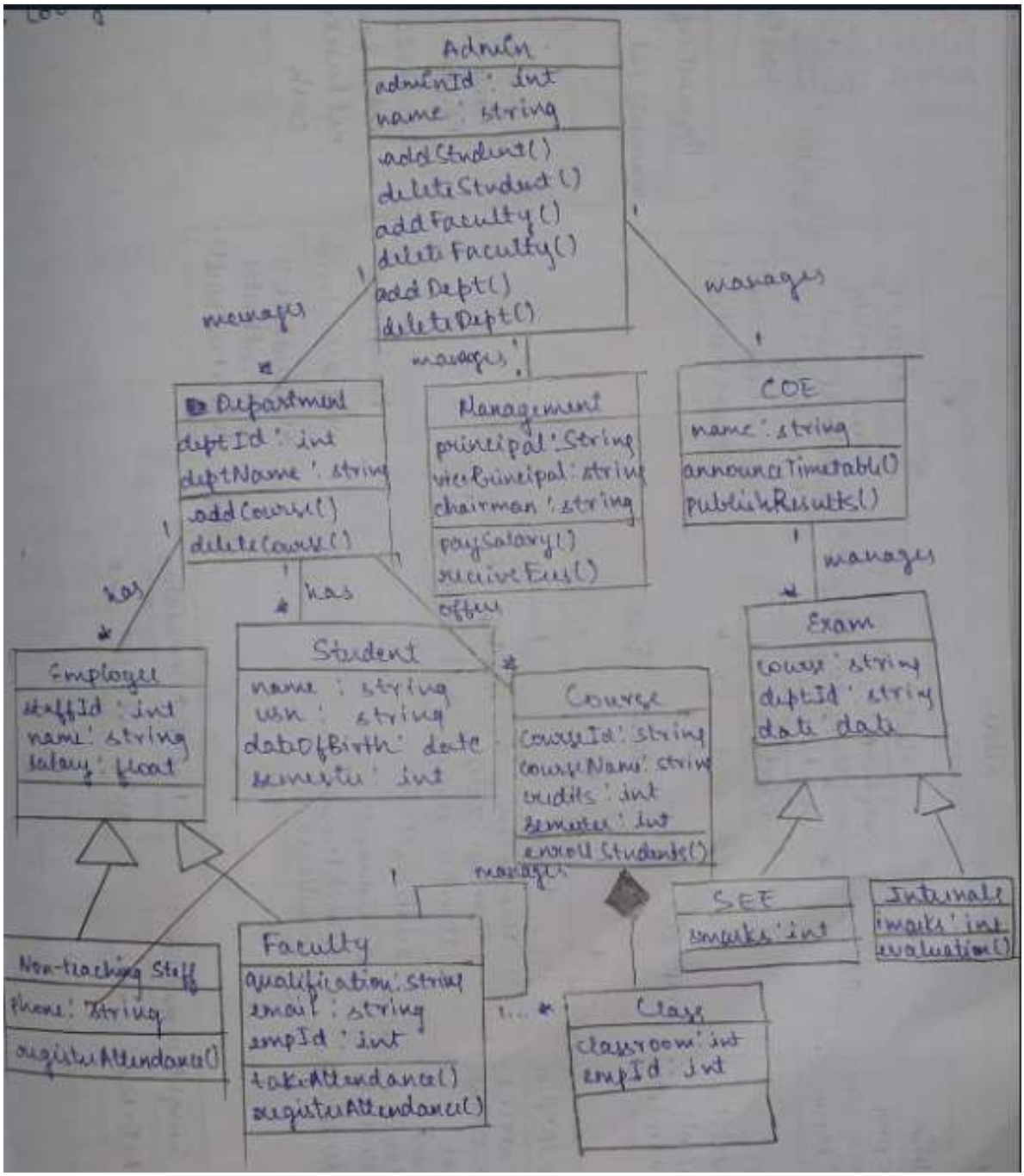
CO4	Ability to conduct practical experiment to solve a given problem using Unified Modeling language.

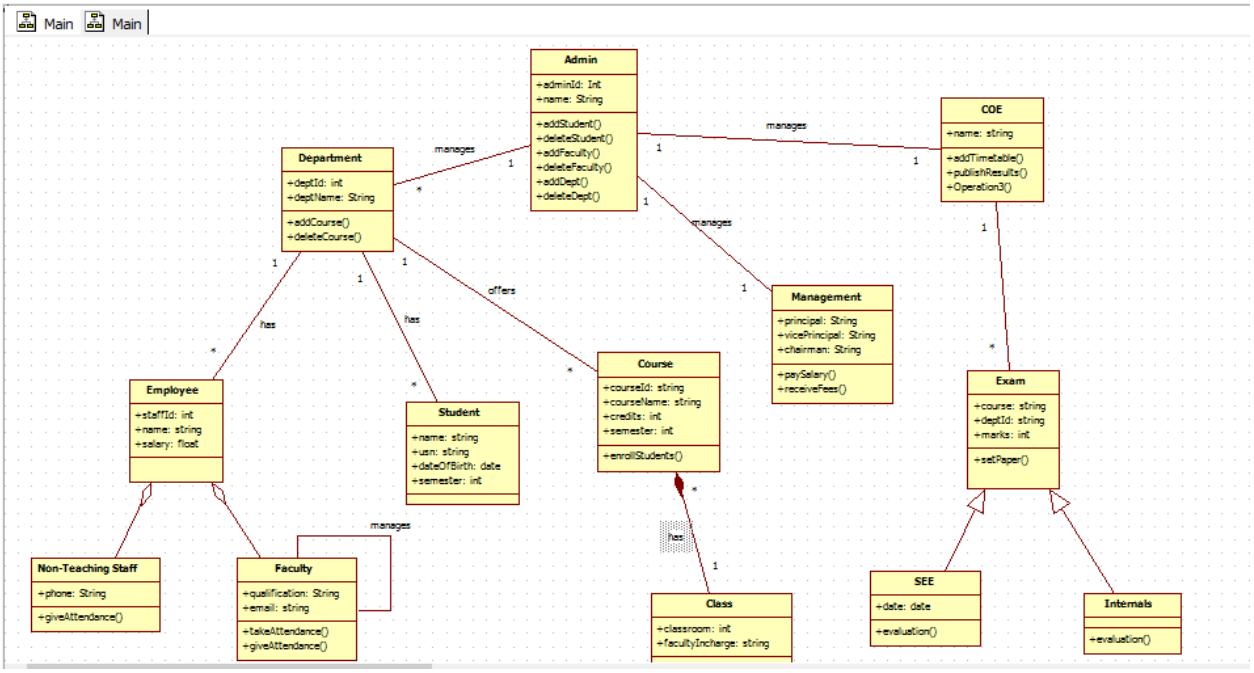
LAB 1:College Information System

1.1 Software Requirement Specification

- College information system has an admin who manages the staff ,students 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 username 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. Internalsand semester end examinations are two 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.

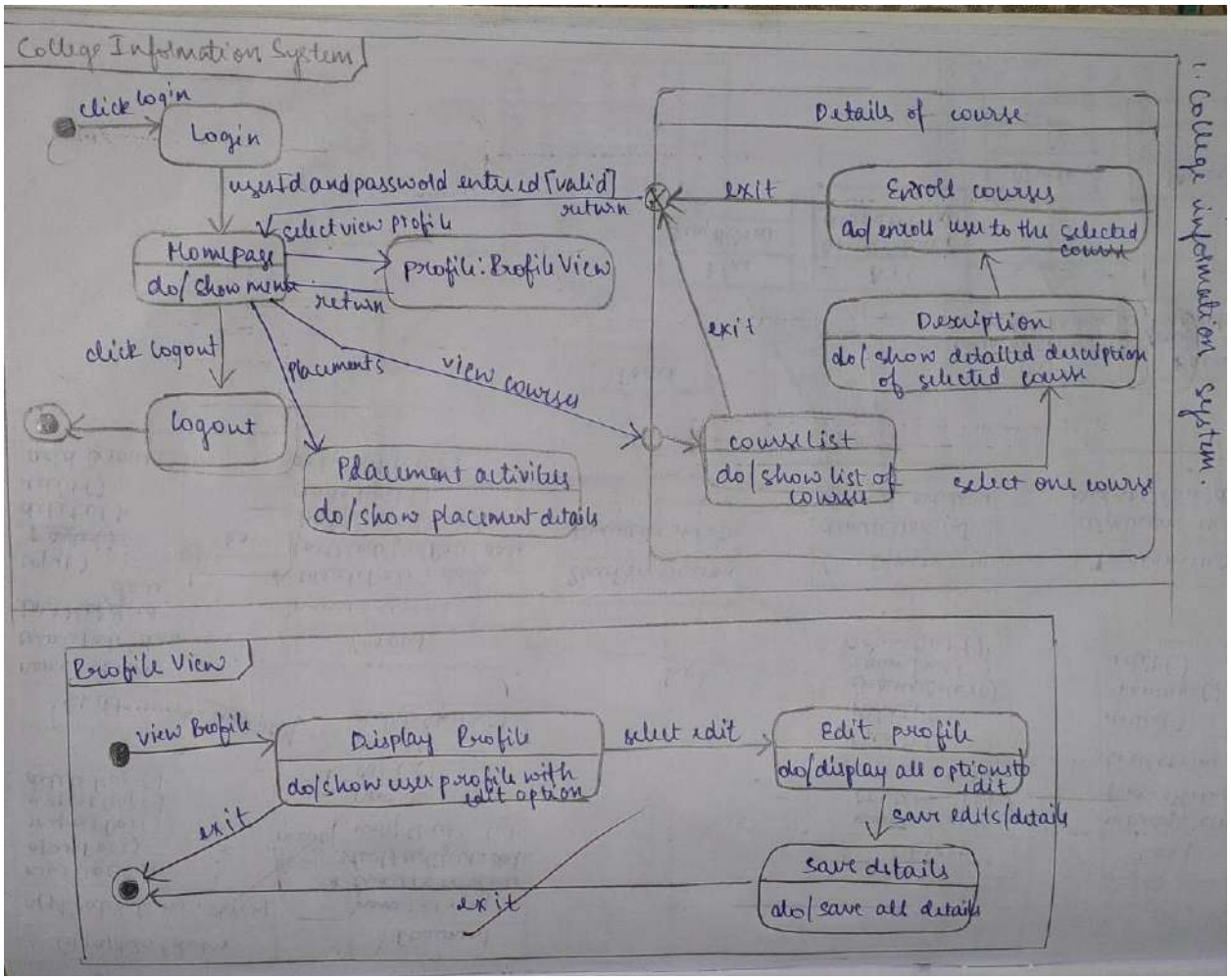
1.2 Advanced Class Diagram

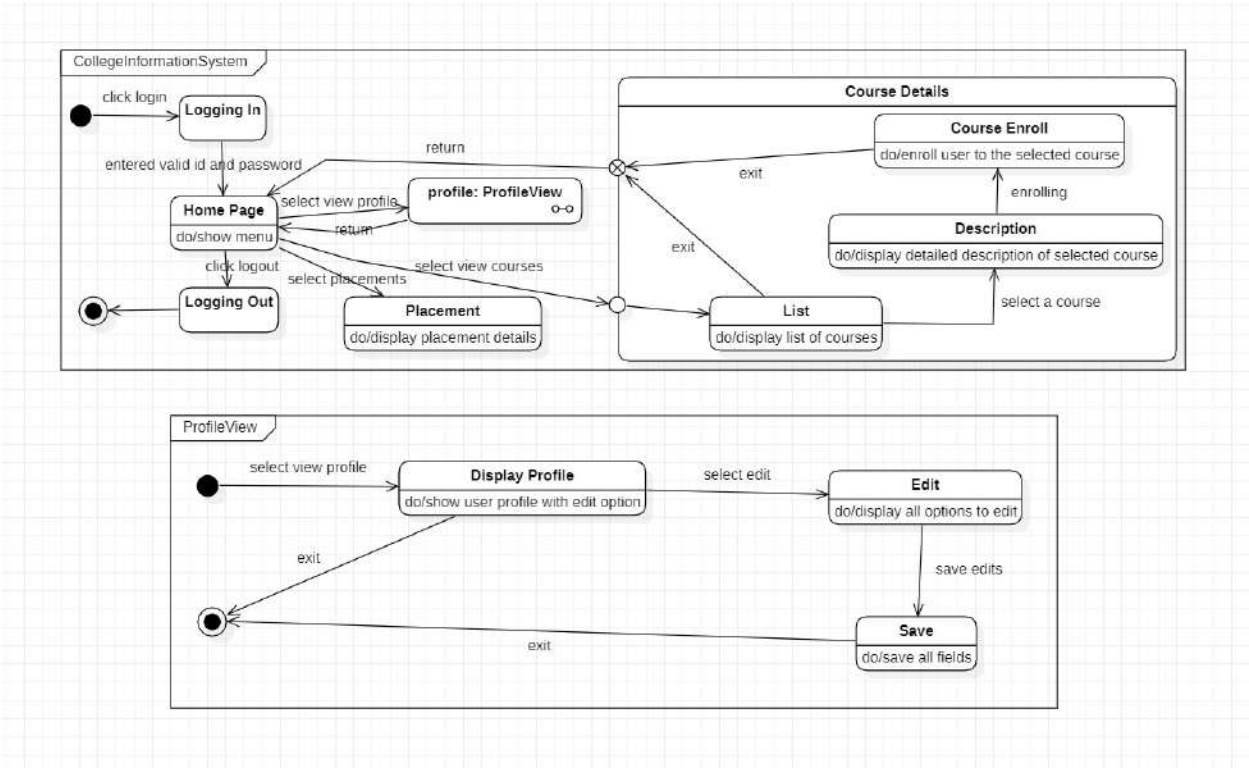




- Admin is associated with the Department, Management and COE. One admin manages Management, COE and many departments in college.
- Department is associated with Employee, Student and Course. Department has many employees, many students and offers many courses.
- Employee class is generalized into non-teaching staff and faculty.Faculty is self associated.
- Because the class cannot exist without courses ,the class is composed inside courses.
- Exam is associated with COE.One COE manages many exams.
- Exams are generalized into SEE and Internals as they use all the attributes of Exam.

1.3 Advance State Diagram

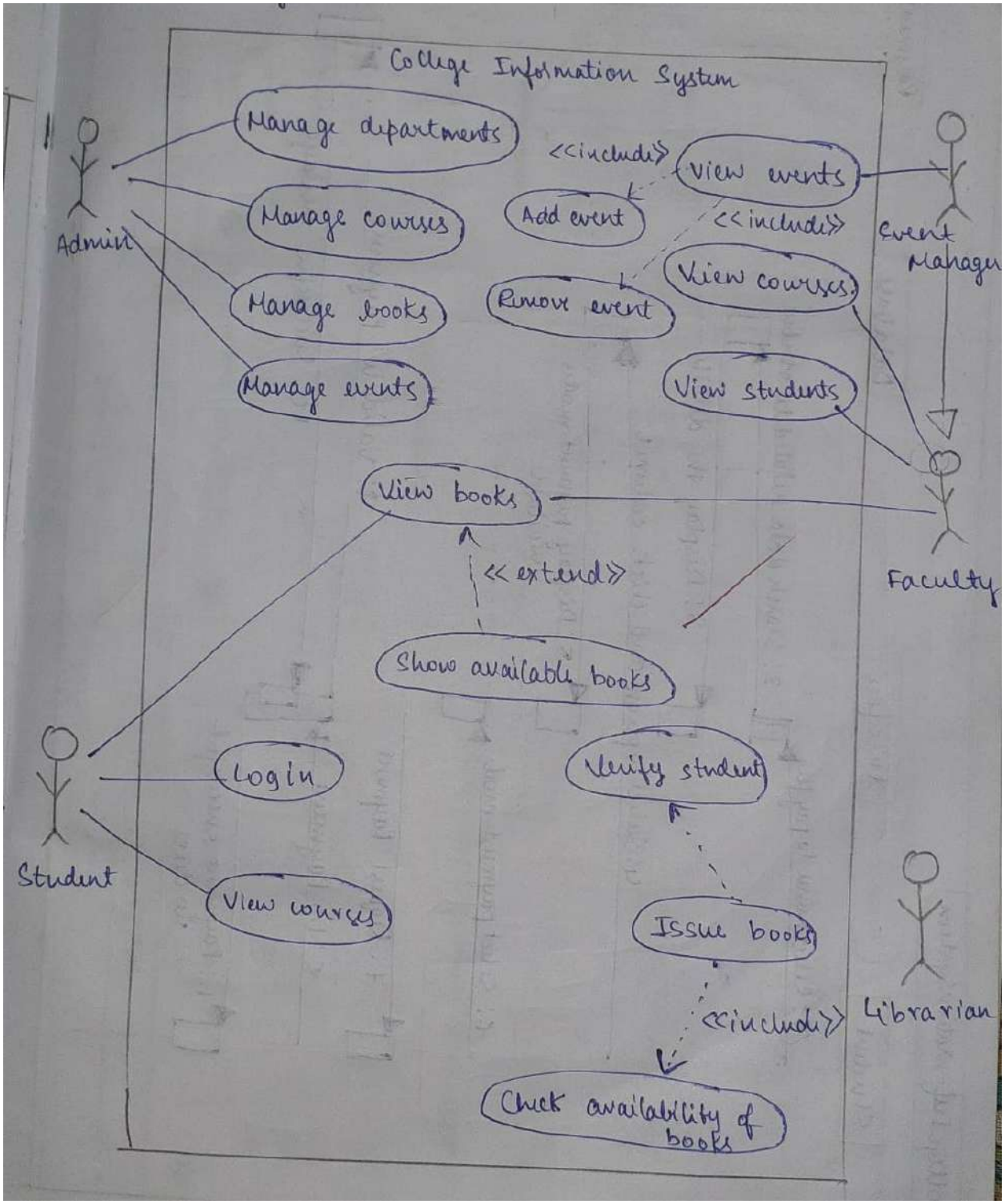


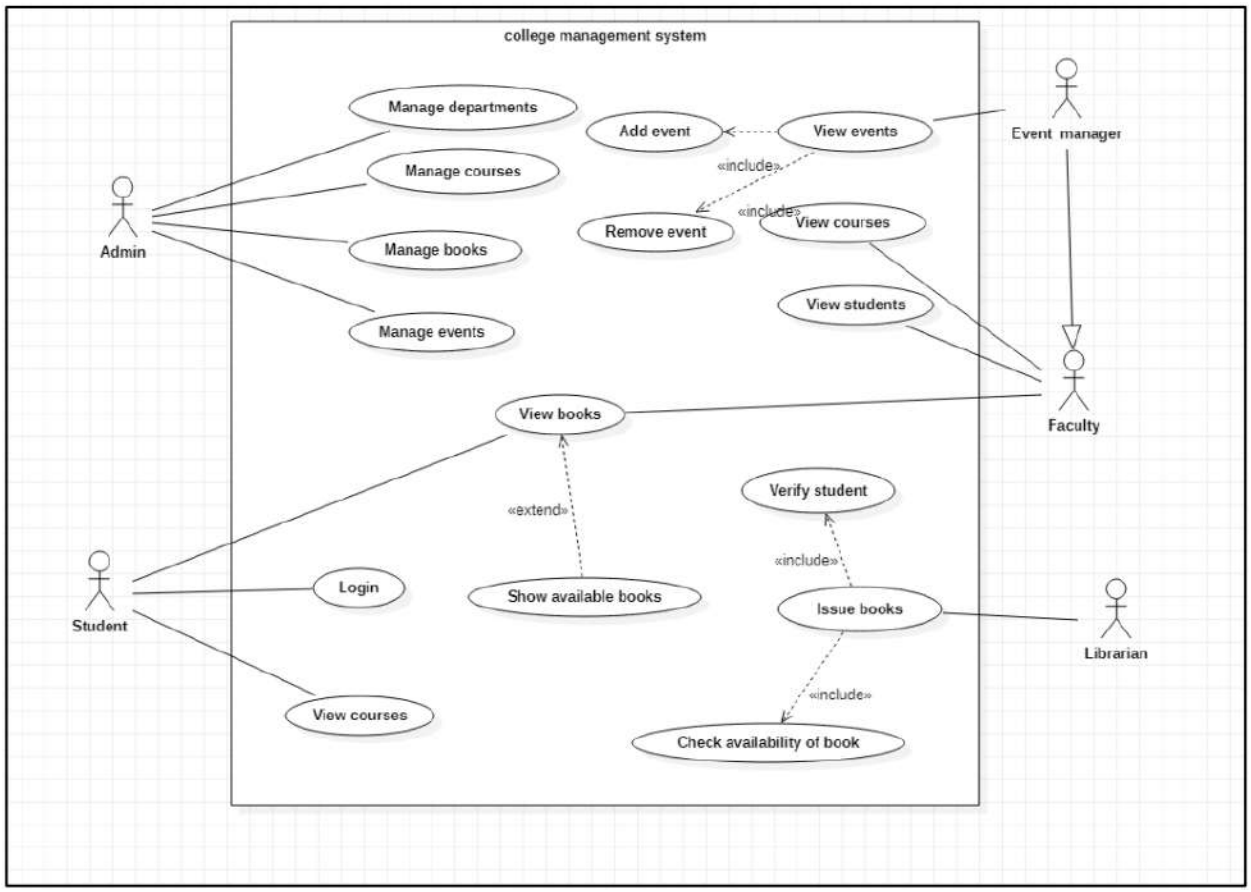


The advanced state diagram contains one nested state and one submachine, which on successful login shows the course details and profileView procedure of the student. It contains initial state and termination state with Courses as a nested state including the required simple states. It also has a submachine state named ProfileView with initial, termination state along with simple states; Display profile, Edit, Save.

1.4 Advanced Use case Diagram

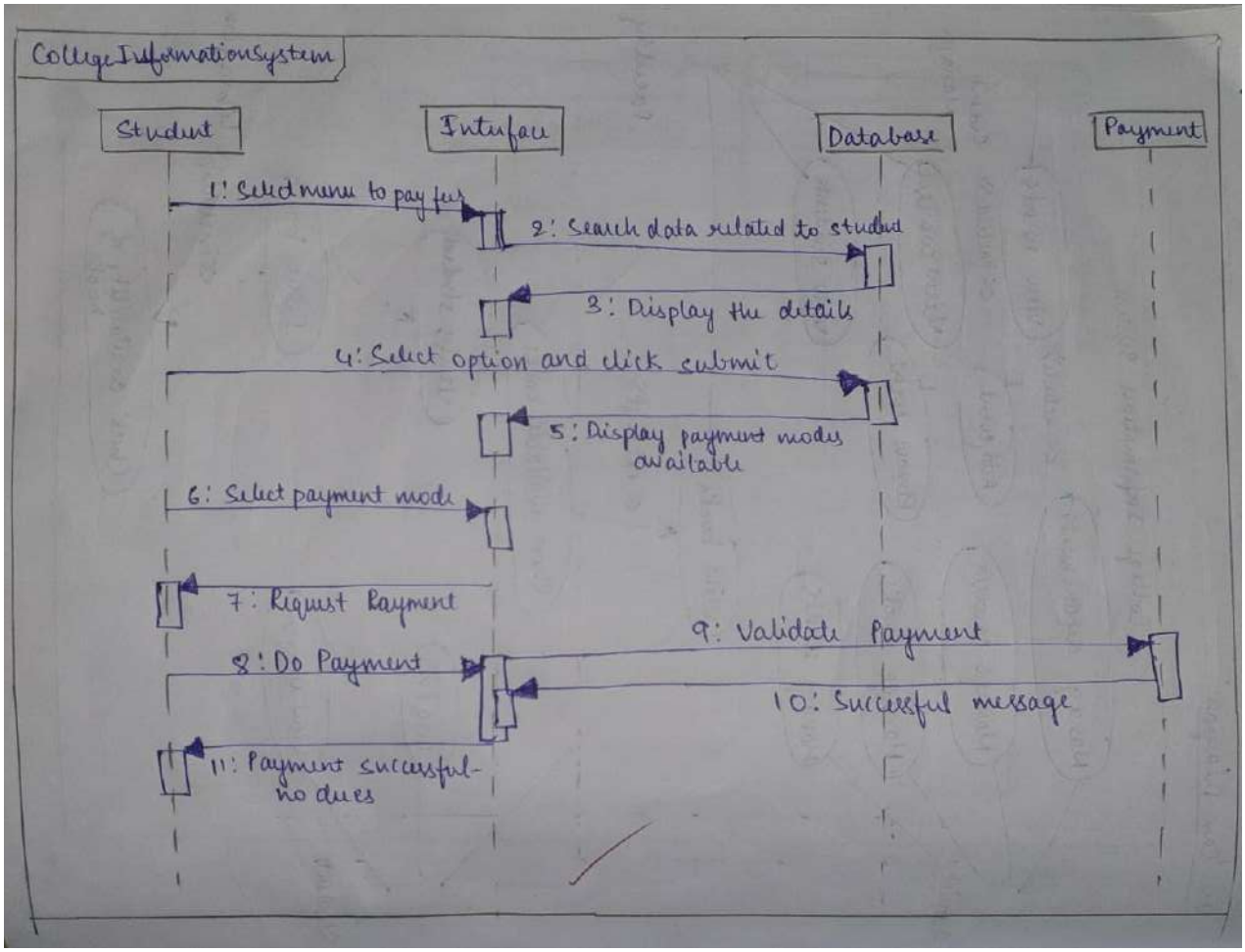
The advanced use case diagram has extra functionalities which includes extends, includes and generalization. The show available books use case extends view books use case, view events use case includes add events and remove events, issue books use case includes verify student and check availability of book.

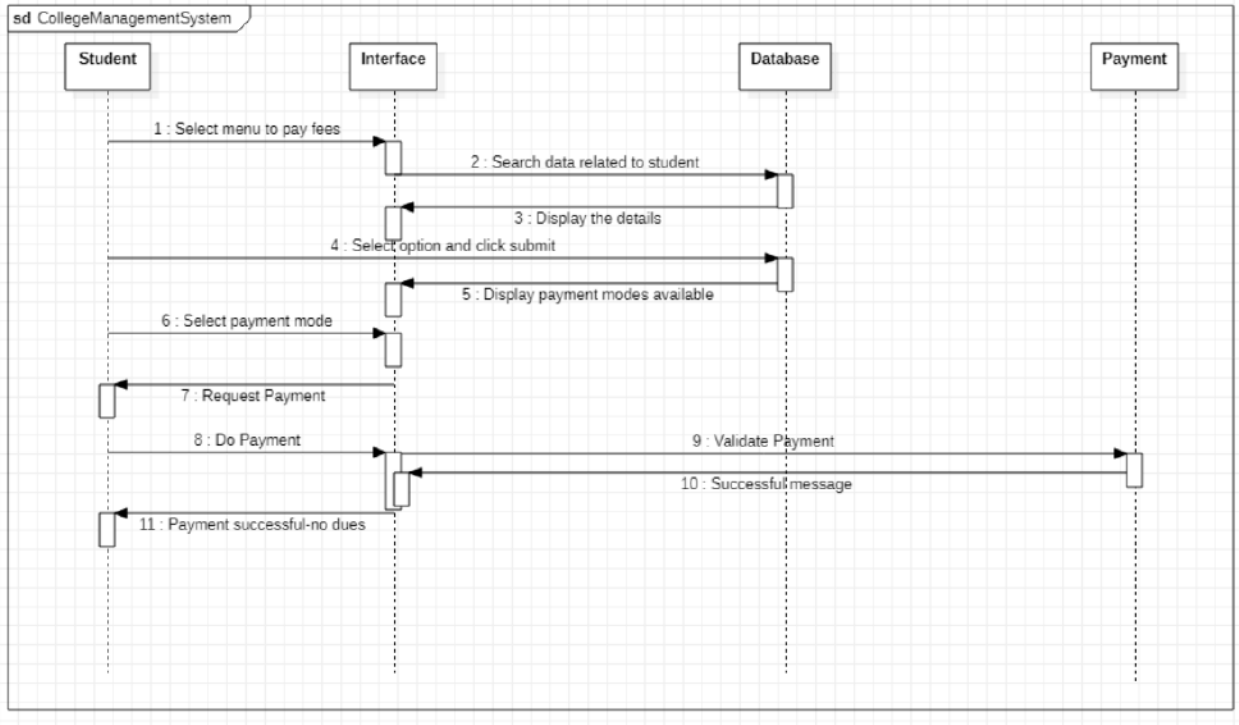

College Information System
Manage departments
Manage courses
Manage books
Manage events
Admin
<<includi>>
Add event
Remove event
View events
<<includi>>
View courses
View students
Event Manager
View books
<<extend>>
Show available books
Faculty
Login
Student
View courses
Verify student
Issue books
<<includi>>
Check availability of books
Librarian



1.5 Advanced Sequence Diagram

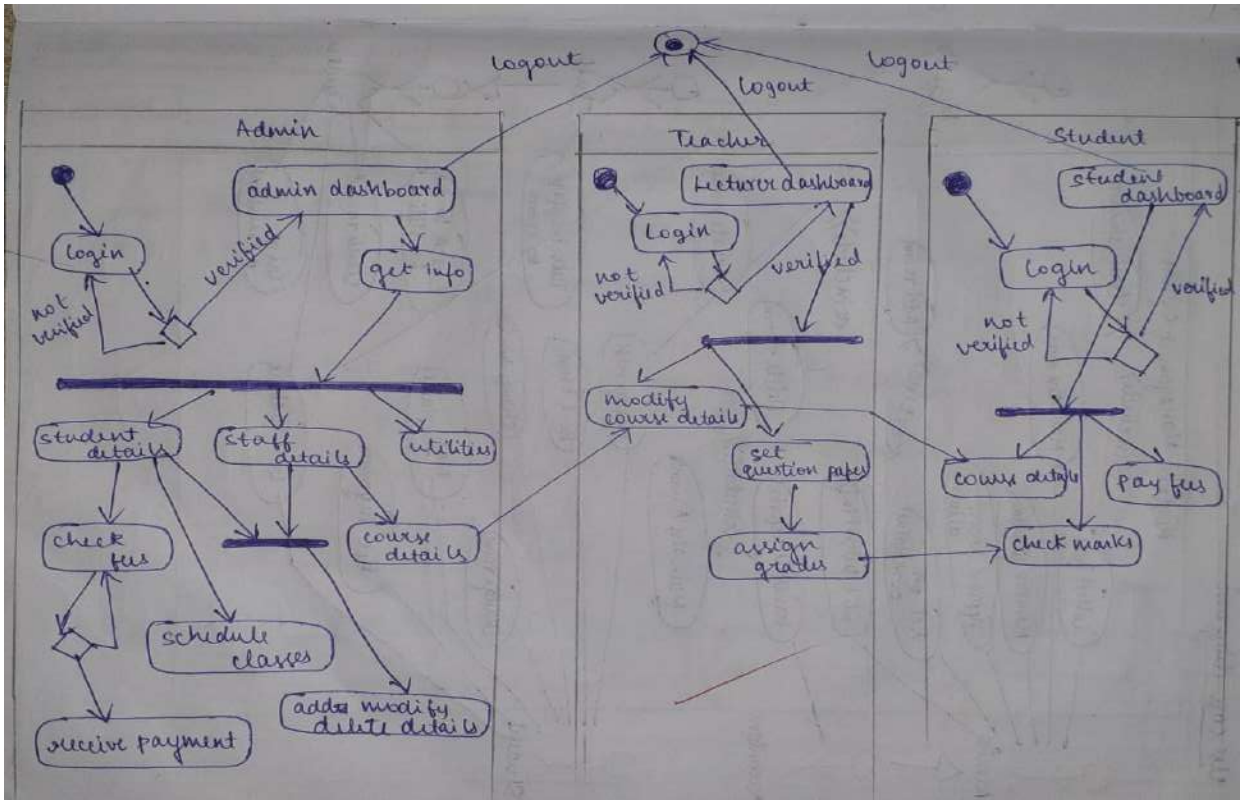
The lifeline is the dotted line and the rectangles represent the period of time the object is executing and is hence called activation. The recursive function of verify is shown by double activation rectangle of verify payment and successful message. The scenario shown here is student logging in and paying fees.

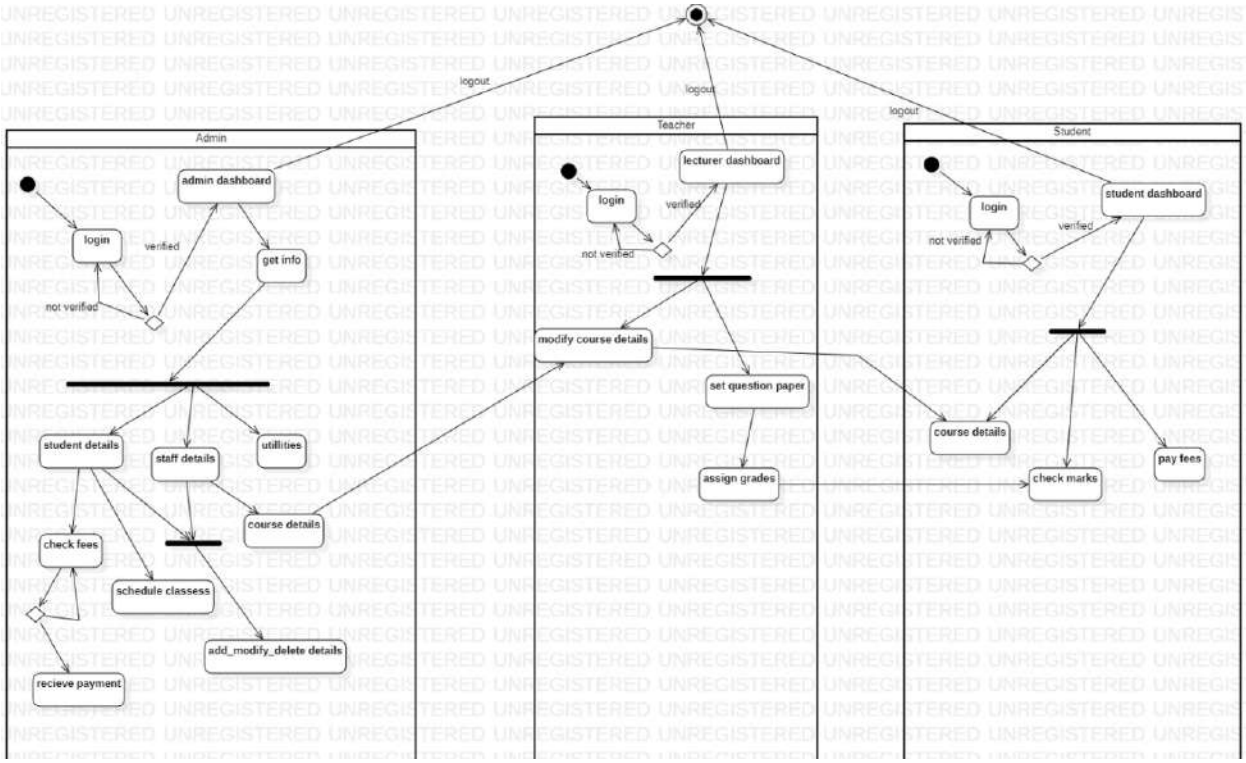

College Information system
Student
Interface
Database
Payment
1: Select menu to pay fee
2: Search data related to student
3: Display the details
4: Select option and click submit
5: Display payment modes available
6: Select payment mode
7: Request Payment
8: Do Payment
9: Validate Payment
10: Successful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



1.6 Advanced Activity Diagram

There are three swimlanes for Admin, Teacher and Student. On successful login, they can see their respective dashboard and on logout the activity is terminated. The admin can check and modify all details. The teacher can set question paper and assign grades. The student can pay fees.


logout
logout
logout
Admin
Teacher
Student
admin dashboard
login
verified
not verified
get info
lecturer dashboard
Login
not verified
verified
student dashboard
Login
not verified
verified
student details
staff details
utilities
modify course details
set question paper
course details
pay fees
check fees
course details
assign grades
check marks
schedule classes
add modify delete details
recieve payment

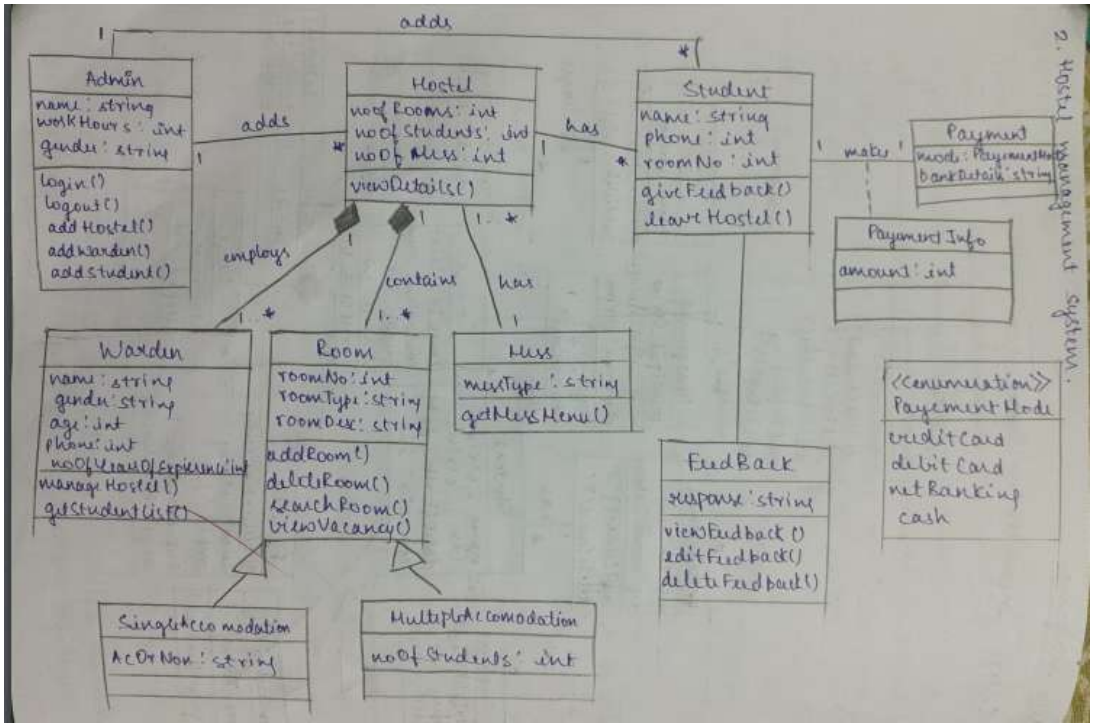

logout
logout
logout
Admin
Teacher
Student
admin dashboard
login
verified
not verified
get info
lecturer dashboard
login
verified
not verified
student dashboard
login
verified
not verified
student details
staff details
utilities
modify course details
set question paper
course details
pay fees
check fees
course details
assign grades
check marks
schedule classess
add_modify_delete details
recieve payment

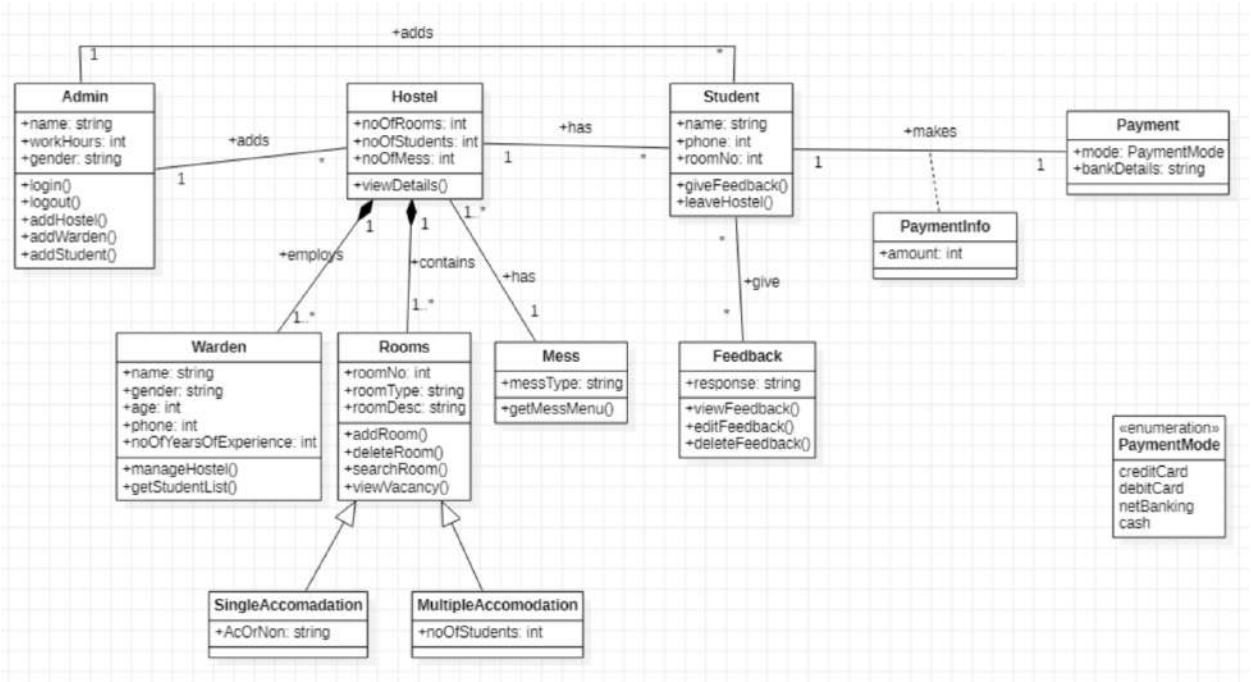
LAB 2: Hostel Management system

2.1 Software Requirement Specification

- Hostel management system has an admin who manages the hostel,allot-es and payment methods.The admin will allocate a room to students according to the section or class. Theadmin will also keep track of the payment made by the 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 allot-es 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 maintained in the database.
- The hostel is categorized into two types I.e boys and girls hostel.Each hostel type has different costs ,warden and name.
- A hostel is made up of mess and rooms.A mess account will also generate. This account has the mess status of the whole month. On the basis of this account monthly charges of a student will be defined.
- The hostel management system will allow renewing the student's registration every year.

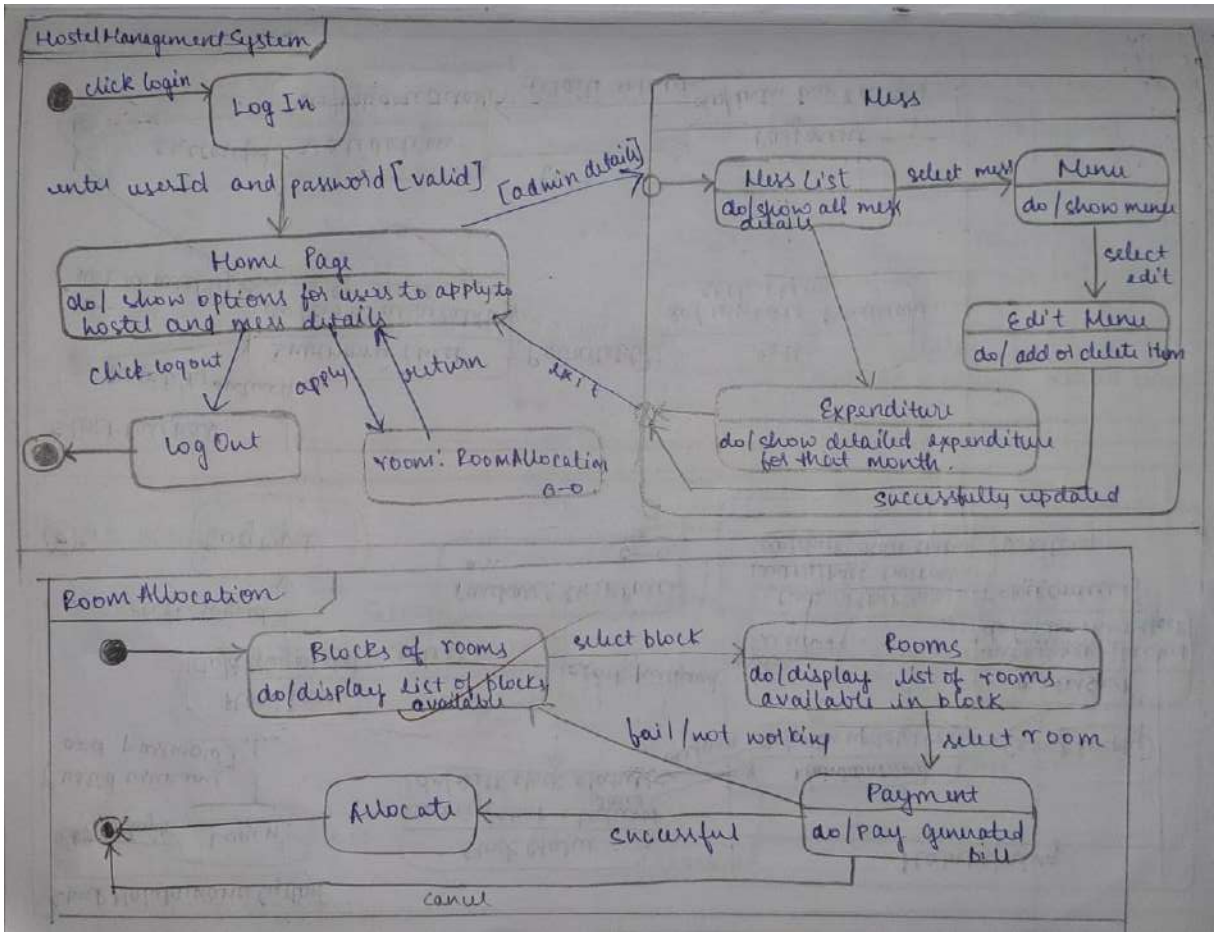
2.2 Advanced Class Diagram

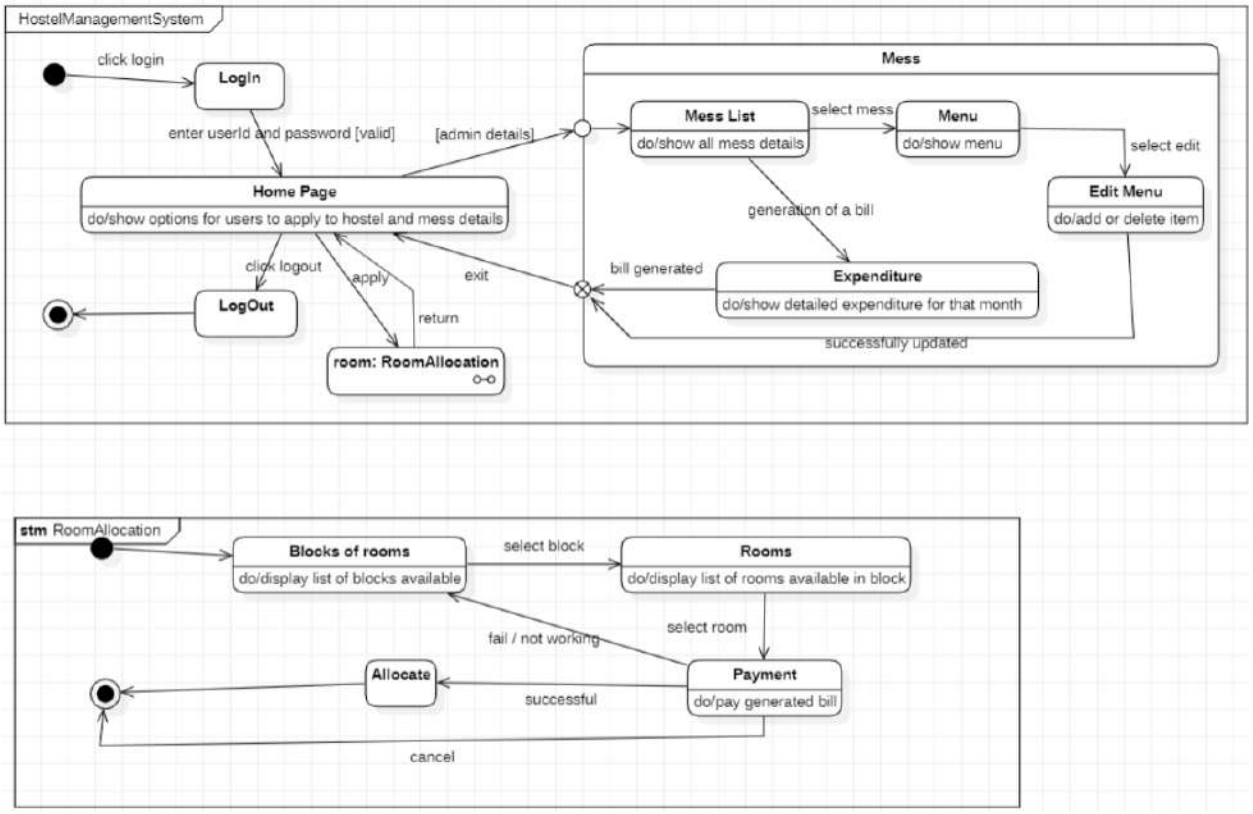




- Here we have Admin class that have association with Hostel class and Student class. One Admin can add many students and hostel.
- The Hostel class is made up of composition of Warden and Rooms. Here composition is used because Warden and Room class cannot exist without Hostel. One hostel can have one or more warden and one or more rooms.
- Many hostels together have a single mess and one hostel has many students.
- Rooms class is generalised as single Accommodation and MultipleAccomodation.
- Students can give feedback and make payement.
- Here payment mode is an enumeration made up of creditCard, debitCard, netBanking and cash.
- PaymentInfo is an association class arising because of the association between student and payment.

2.3 Advanced State Diagram

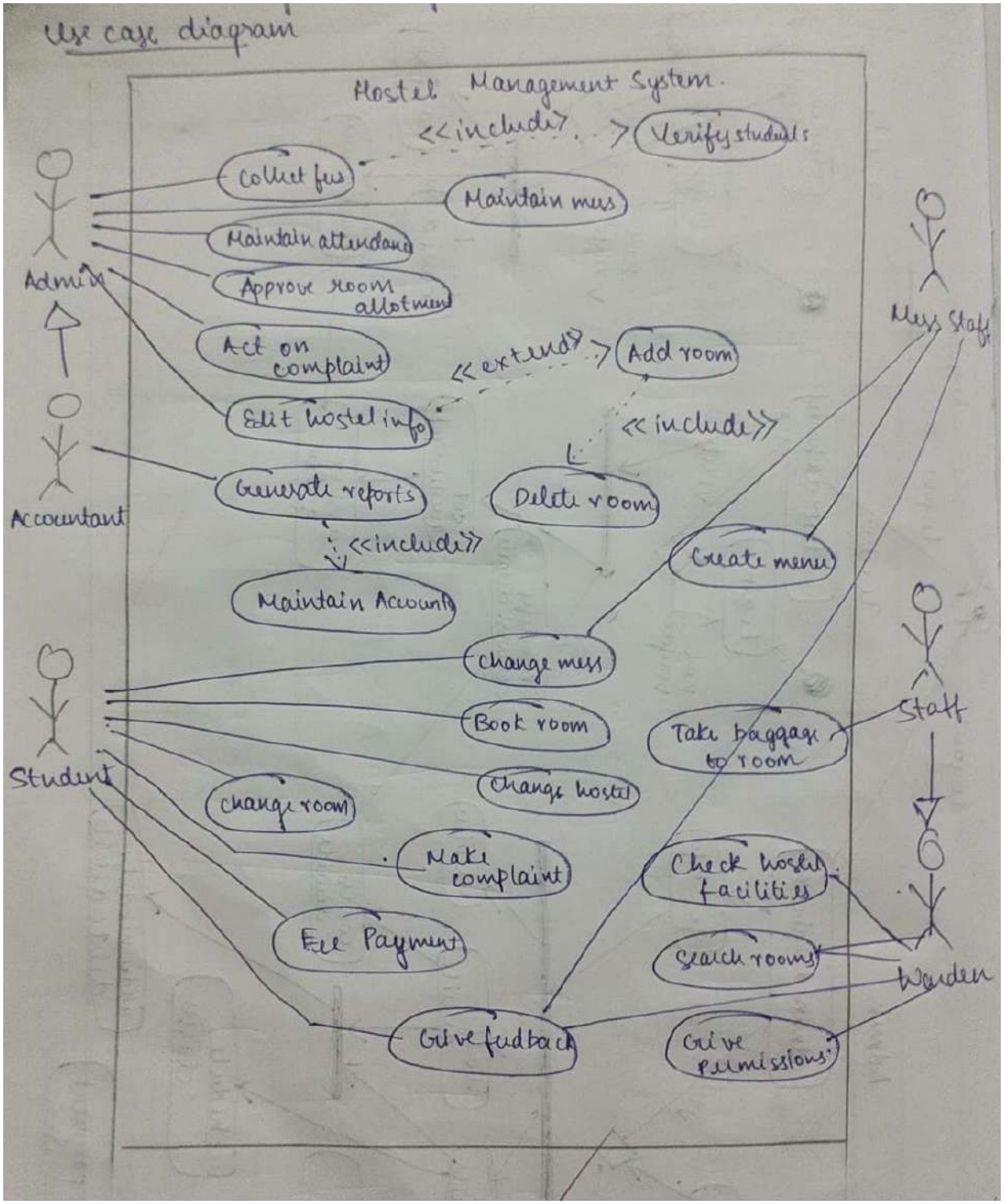

Hostel Management System
click login
Log In
enter userId and password [valid]
[admin details]
Home Page
do/ show options for user to apply to hostel and mess details
click logout
apply
return
exit
Log Out
room: RoomAllocation
Mess
Mess List
do/ show all mess details
select mess
Menu
do/ show menu
select edit
Edit Menu
do/ add or delete item
Expenditure
do/ show detailed expenditure for that month.
successfully updated
Room Allocation
Blocks of rooms
do/display list of blocks available
select block
Rooms
do/display list of rooms available in block
fail/not working
select room
Payment
do/pay generated bill
Allocate
successful
cancel

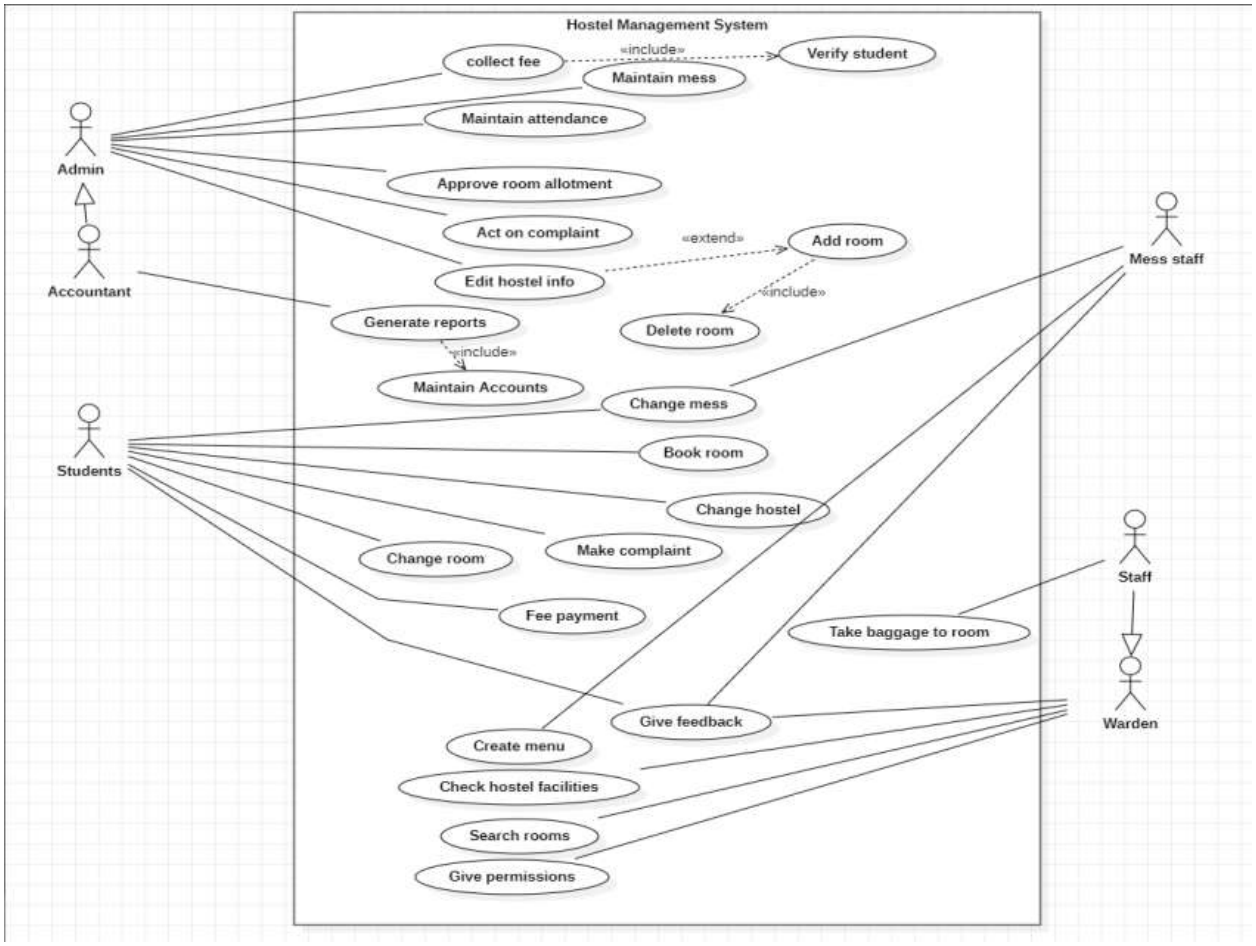


The above state diagram gives the movement of states in allotting a room to a student and admin can change mess menu.

2.4 Advanced Use case Diagram

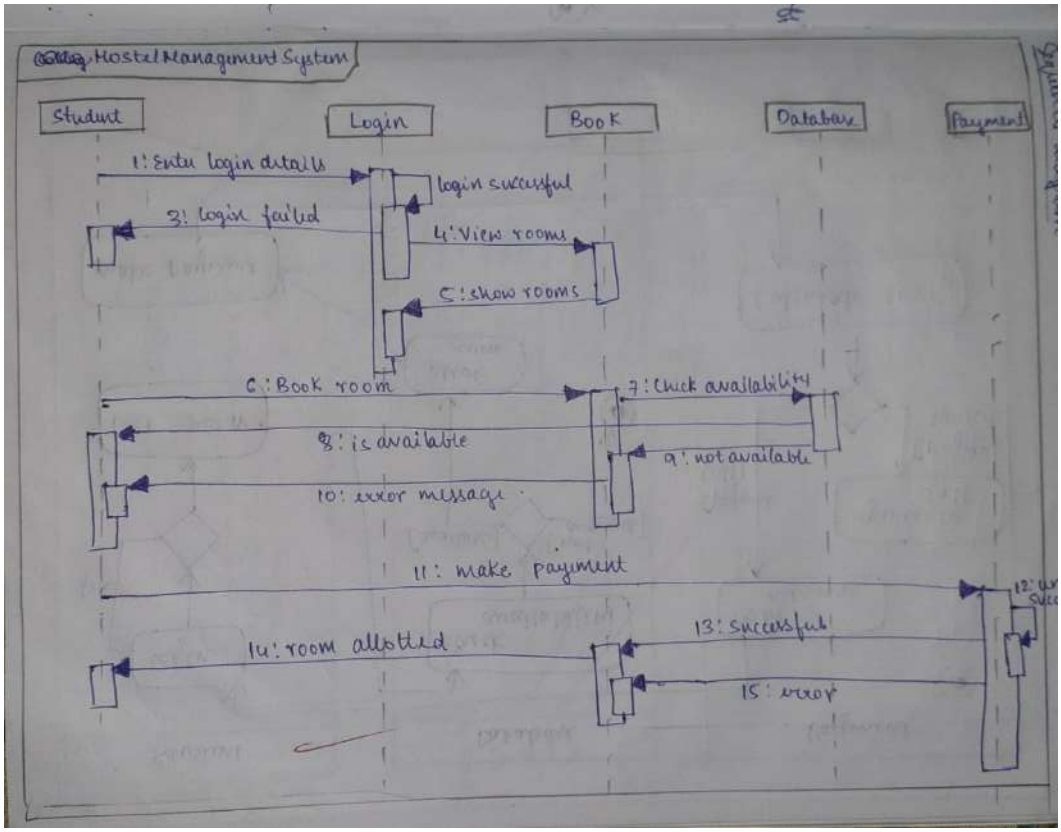
The advanced use case diagram has extra functionalities which includes extends, includes and generalization. The edit hostel info use case extends add room use case, collect fee use case includes verify student, add room use case includes delete room use case.

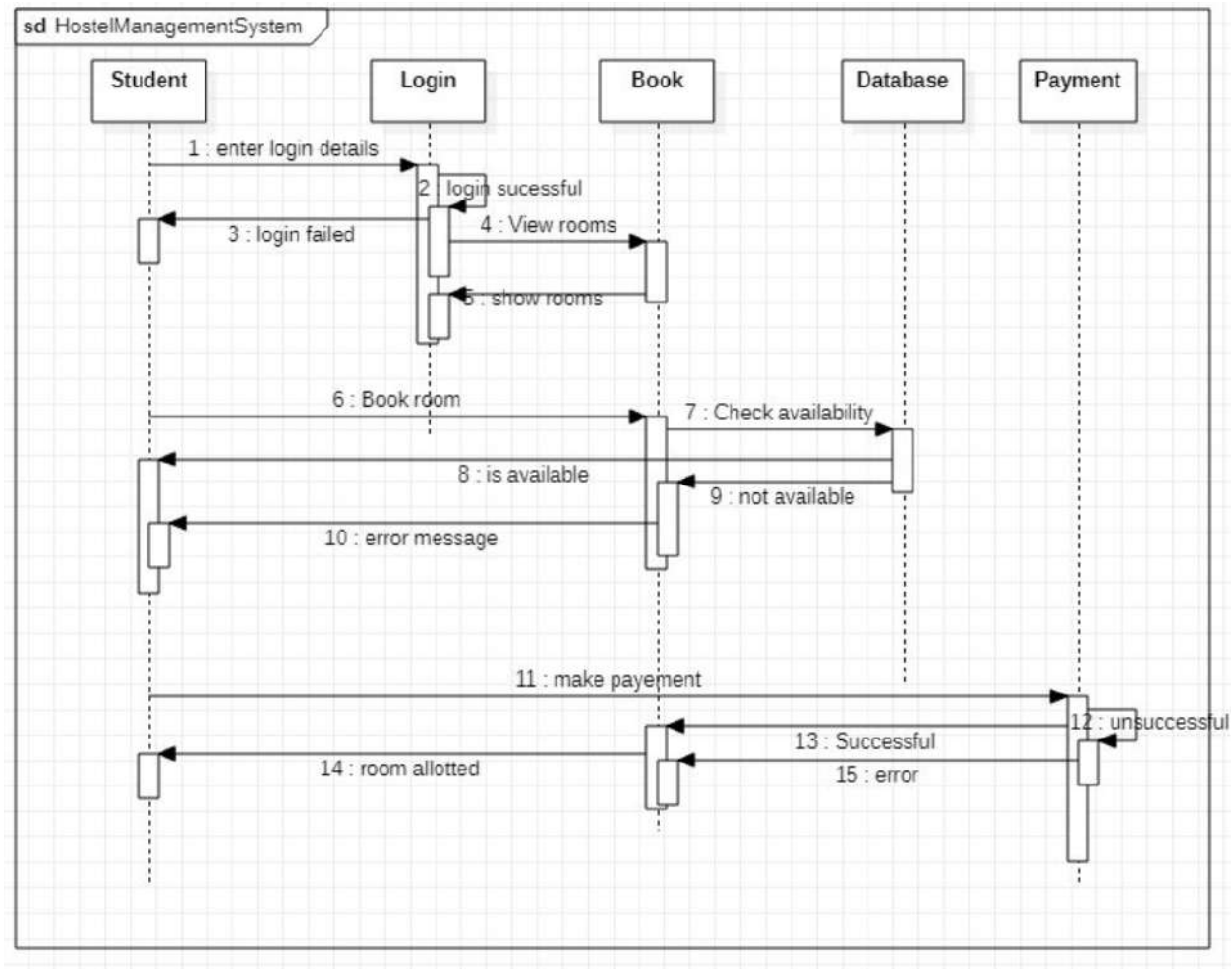

Use case diagram
Hostel Management System
<<include>>
Verify student
Collect fee
Maintain mess
Maintain attendance
Approve room allotment
Act on complaint
Edit hostel info
<<extend>>
Add room
<<include>>
Delete room
Generate reports
<<include>>
Maintain Accounts
Create menu
change mess
Book room
Change hostel
Change room
Make complaint
Pay Payment
Give feedback
Take baggage to room
Check hostel facilities
Search rooms
Give permissions
Admin
Accountant
Student
Mess Staff
Staff
Warden



2.5 Advanced Sequence Diagram

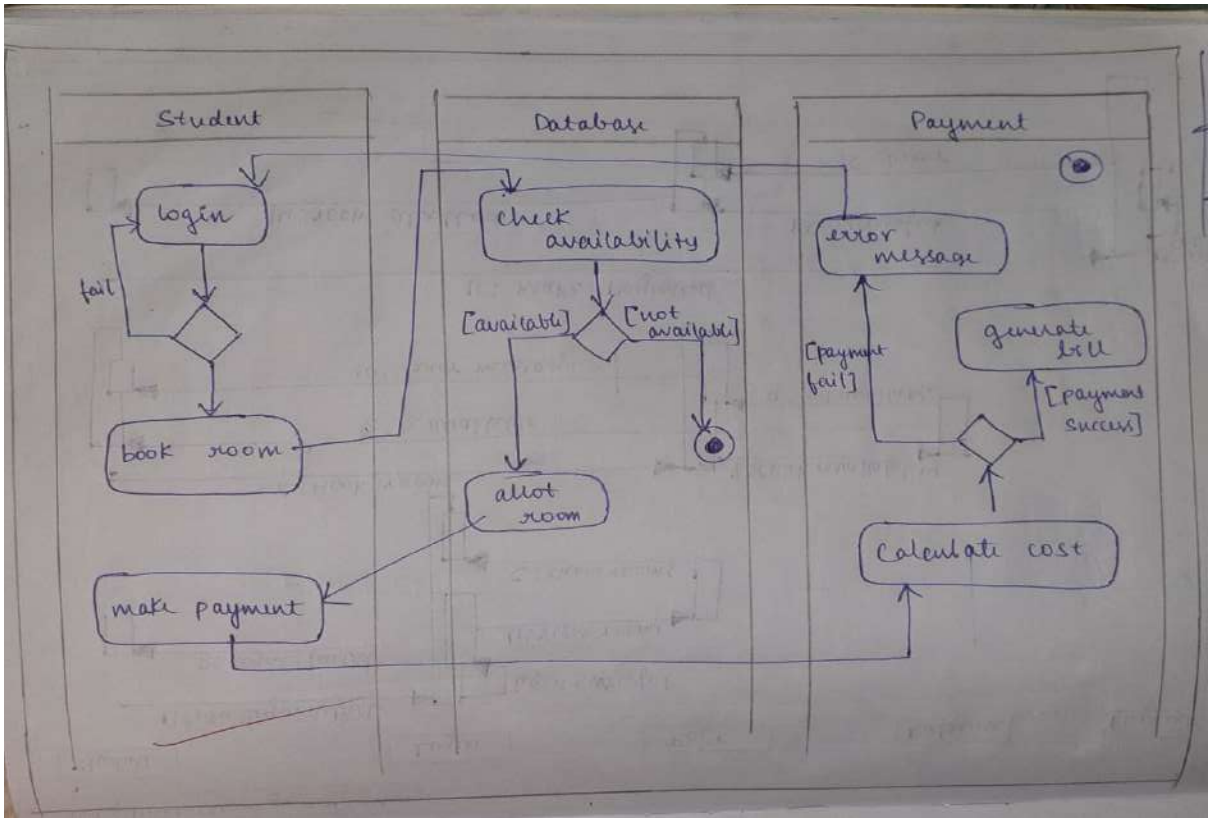
The lifeline is the dotted line and the rectangles represent the period of time the object is executing and is hence called activation. The scenario shown here is a student logging in into system, booking a room and making payment.If the payment is successful the room is alloted.

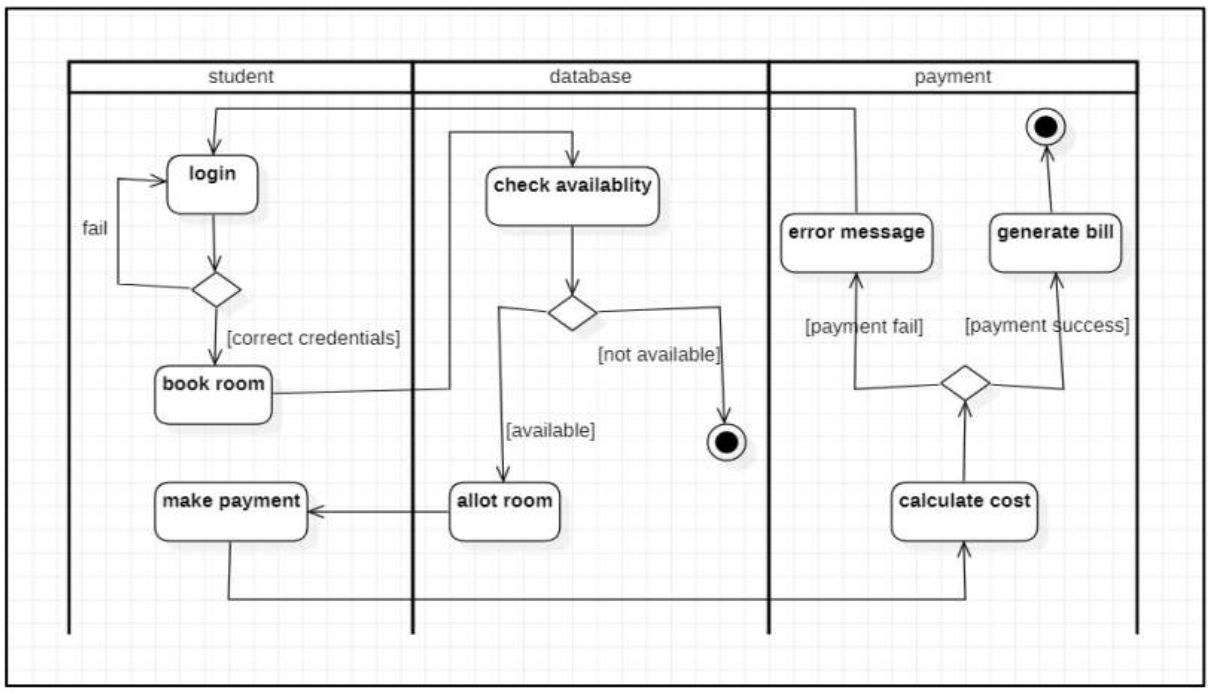

Hostel Management System
Student
Login
Book
Databas
Payment
1: Enter login details
login succesful
3: login failed
4: View rooms
5: show rooms
6: Book room
7: check availability
8: is available
9: not available
10: error message
11: make payment
13: successful
14: room allotted
15: error



2.6 Advanced Activity Diagram

The advanced activity diagram starts from initiation and then in the student swimlane, student login activity where a signal is sent to the network for request validation and upon confirmation the control flows to profile and then book room activity. There are three swimlanes namely student, database, payment where validate student, update database and confirm payment respectively. Then the control flows to the home page and then termination activities.


Student
Database
Payment
login
fail
book room
check availability
[available]
[not available]
allot room
make payment
error message
[payment fail]
generate bill
[payment success]
Calculate cost

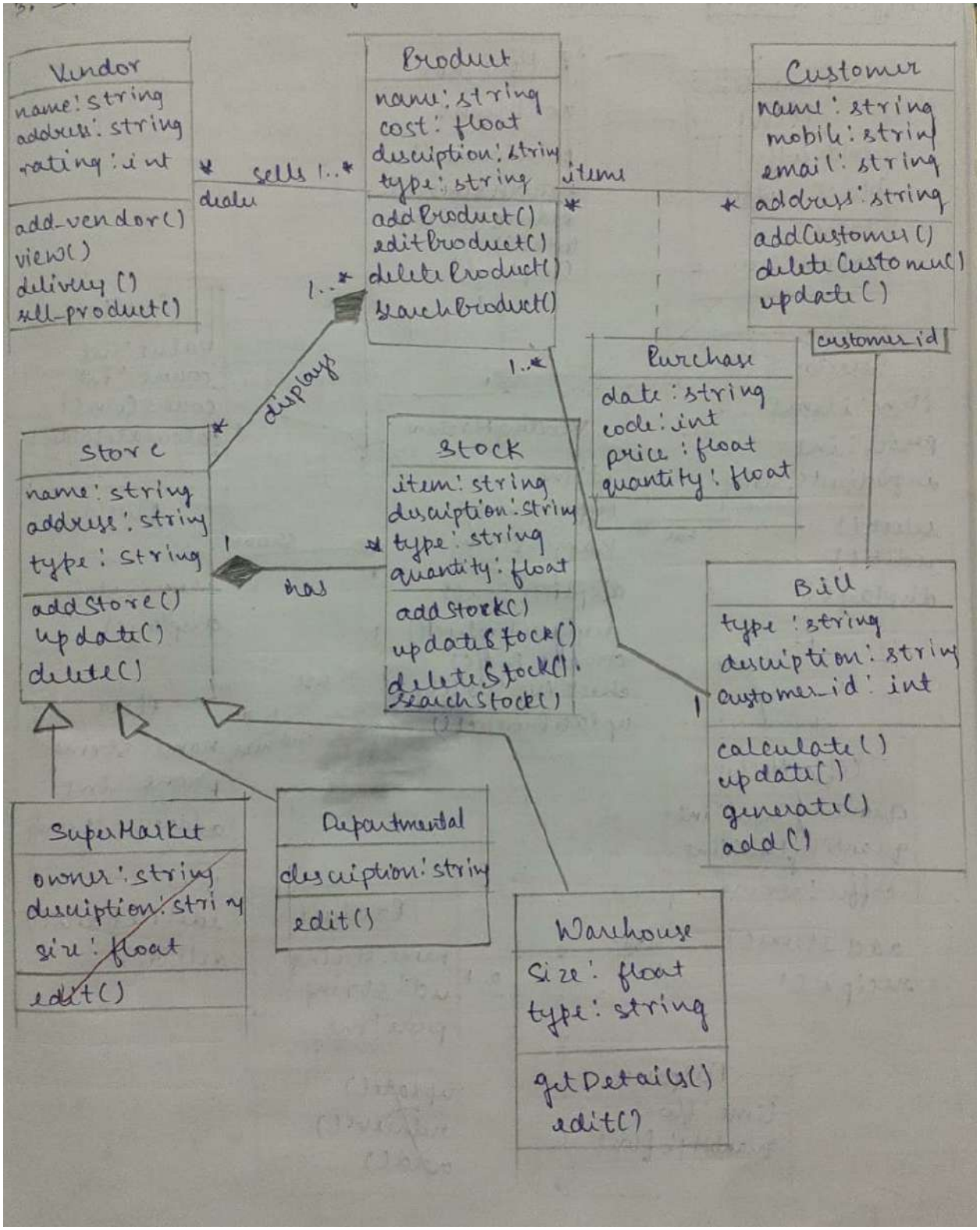

student
database
payment
login
fail
[correct credentials]
book room
check availablity
[not available]
[available]
make payment
allot room
error message
[payment fail]
generate bill
[payment success]
calculate cost

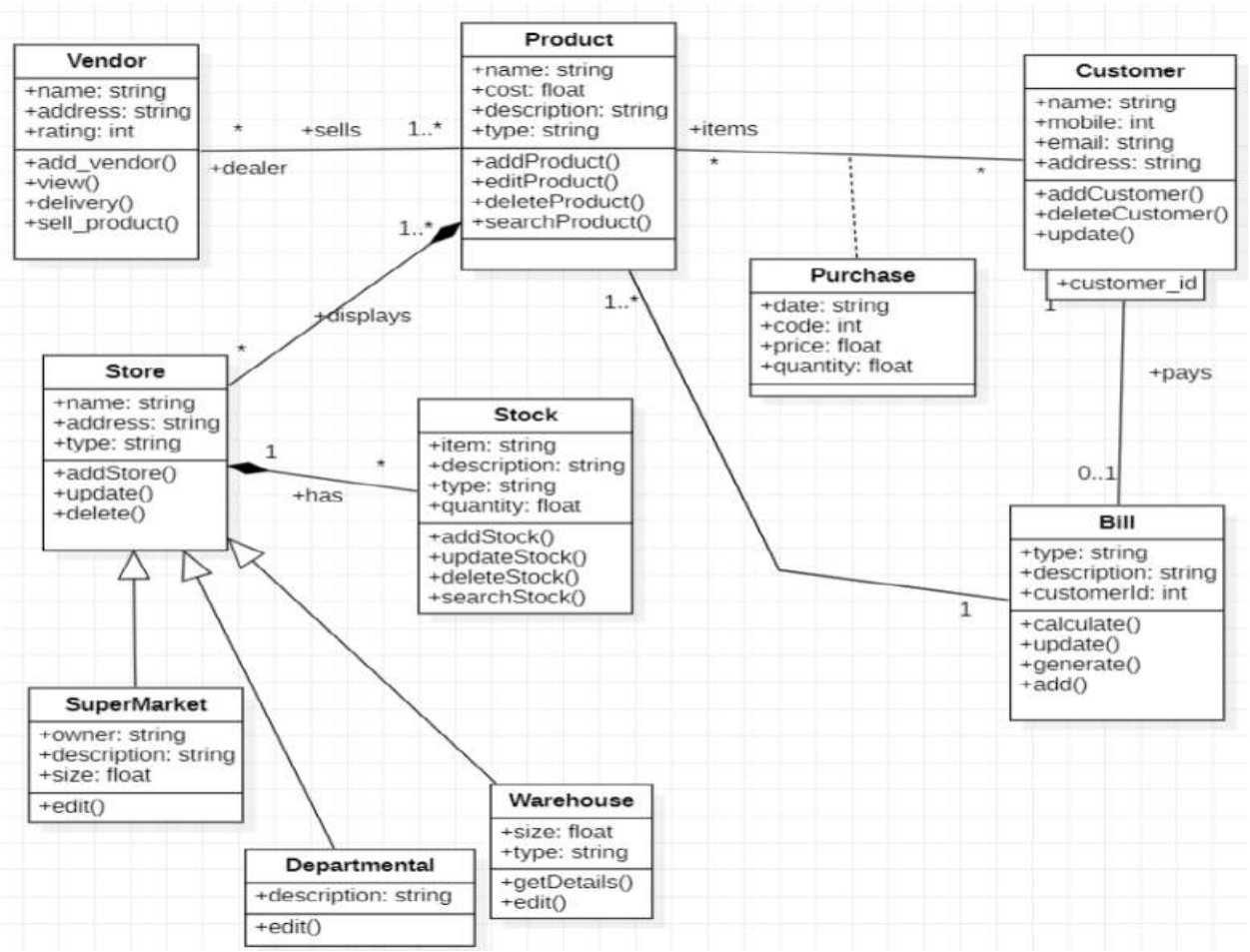
LAB 3:Stock Management System

3.1 Software Requirement Specification

- The customer can purchase one or more products 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 separately,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 products to the organization.
- Vendors consist of details such as vendor name, address, email id, sales tax number etc. This Information 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 store id,name,address and type are used to locate any product.The stores can be of many types.Some of them are departmental stores,supermarkets and warehouses where the products are kept for display.

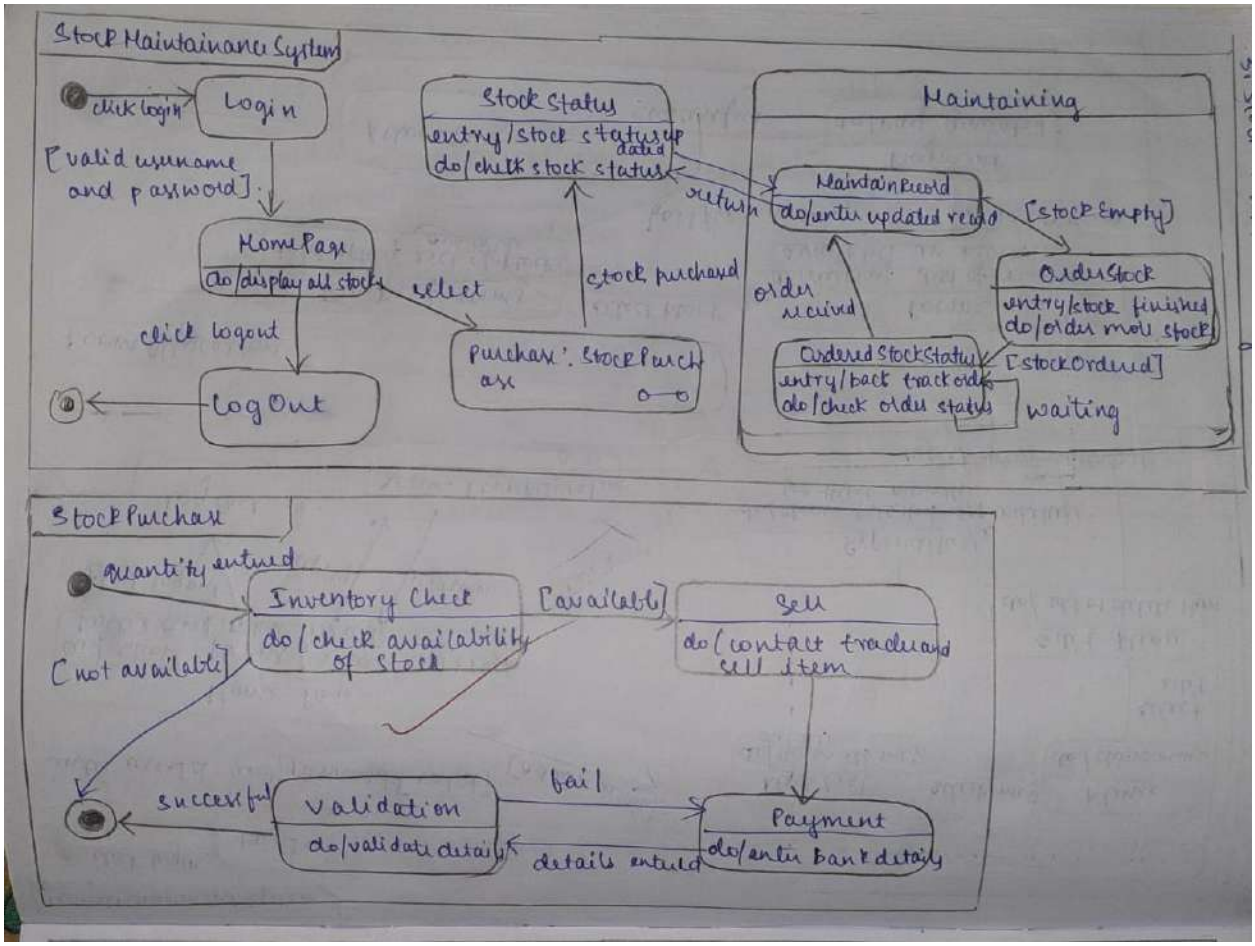
3.2 Advanced Class Diagram

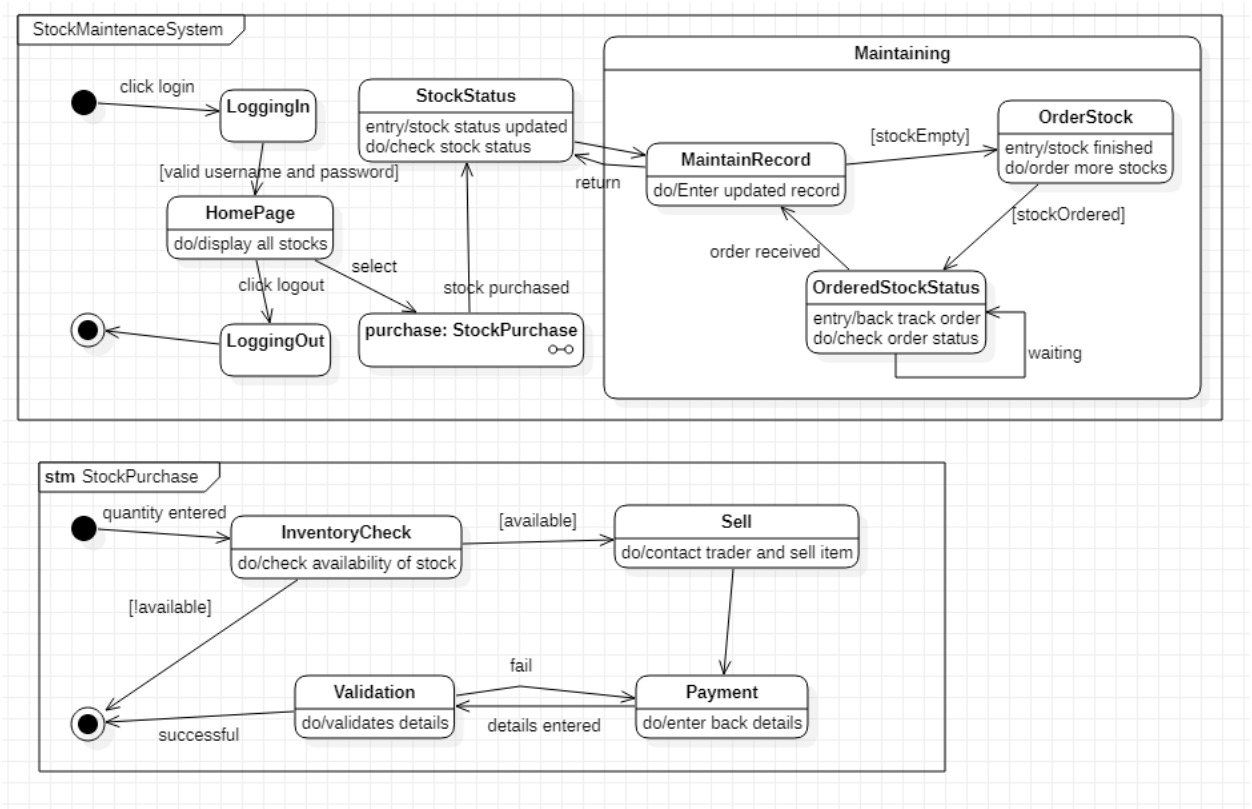

Vendor
name: string
address: string
rating: int
add_vendor()
view()
delivery()
sell_product()
sells
Product
name: string
cost: float
discription: string
type: string
addProduct()
editProduct()
deleteProduct()
searchProduct()
items
Customer
name: string
mobile: string
email: string
address: string
addCustomer()
deleteCustomer()
update()
customer_id
displays
Purchase
date: string
code: int
price: float
quantity: float
Store
name: string
address: string
type: string
addStore()
update()
delete()
has
Stock
item: string
discription: string
type: string
quantity: float
addStock()
updateStock()
deleteStock()
searchStock()
Bill
type: string
discription: string
customer_id: int
calculate()
update()
generate()
add()
SuperMarket
owner: string
discription: string
size: float
edit()
Departmental
discription: string
edit()
Warehouse
Size: float
type: string
getDetails()
edit()



- Vendor deals with a variety of products which are displayed (or stored) in stores.
- A store is generalized to be a superMarket, departmental store or a warehouse.
- A store maintains a stock of the product. The type of association is composition because the stock cannot exist without the store.
- A customer can buy one or more products. We have an association class purchase because of this association.
- To reduce the multiplicity of the association between the customer and bill we have used the qualifier customer id.

3.3 Advanced State Diagram

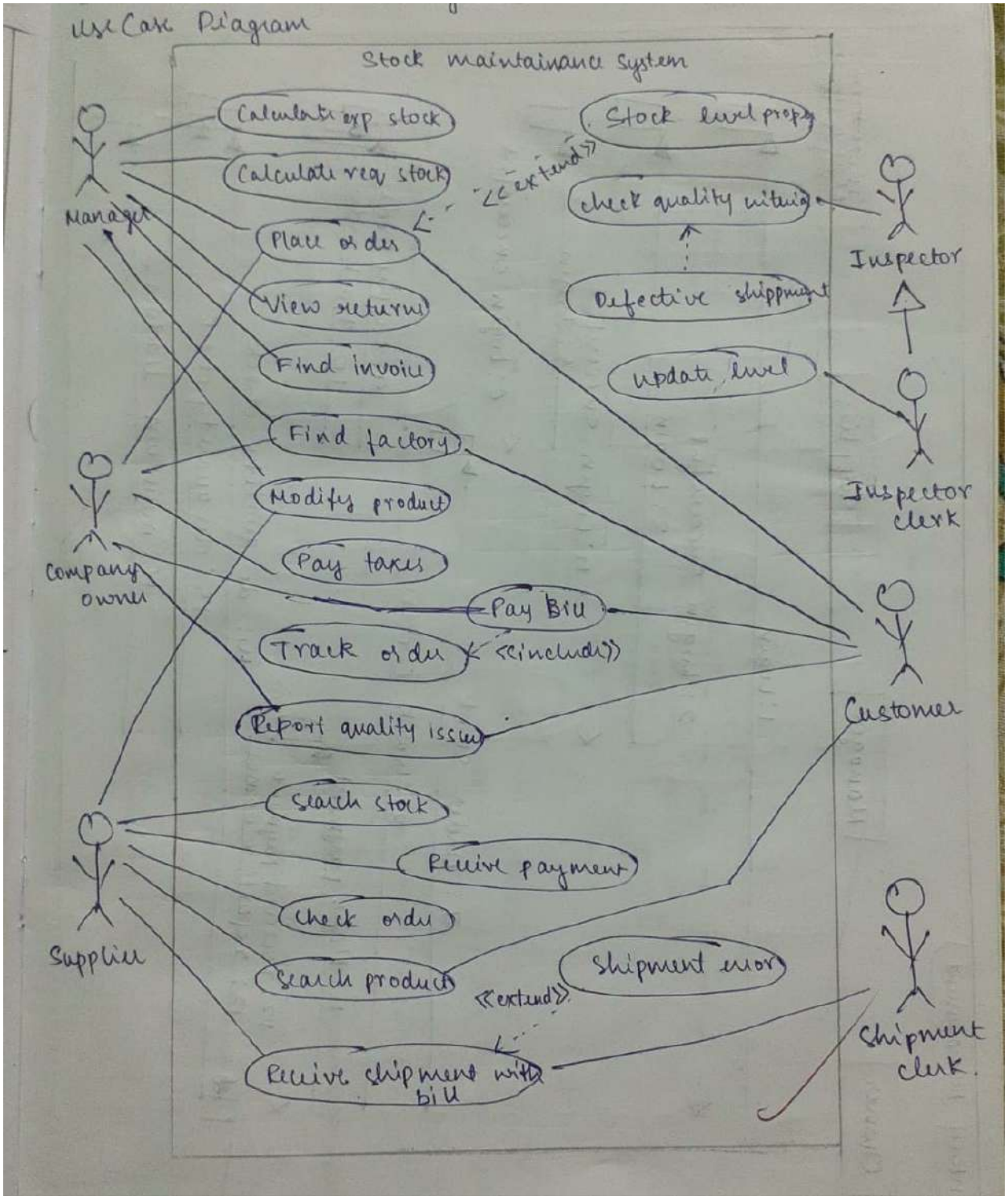

Stock Maintainance System
click login
Login
[valid username and password]
HomePage
do/display all stocks
select
click logout
LogOut
Purchase Stock Purchase
stock purchased
Stock Status
entry/stock status up dated
do/chek stock status
return
Maintaining
Maintain Record
do/enter updated record
[stock Empty]
Order Stock
entry/stock finished
do/order more stock
order received
Ordered Stock Status
entry/back track order
do/check order status
waiting
[stock Ordered]
Stock Purchase
quantity entered
Inventory Check
do/check availability of Stock
[not available]
[available]
Sell
do/contact trader and sell item
successful
Validation
do/validate details
fail
details entered
Payment
do/enter Bank details

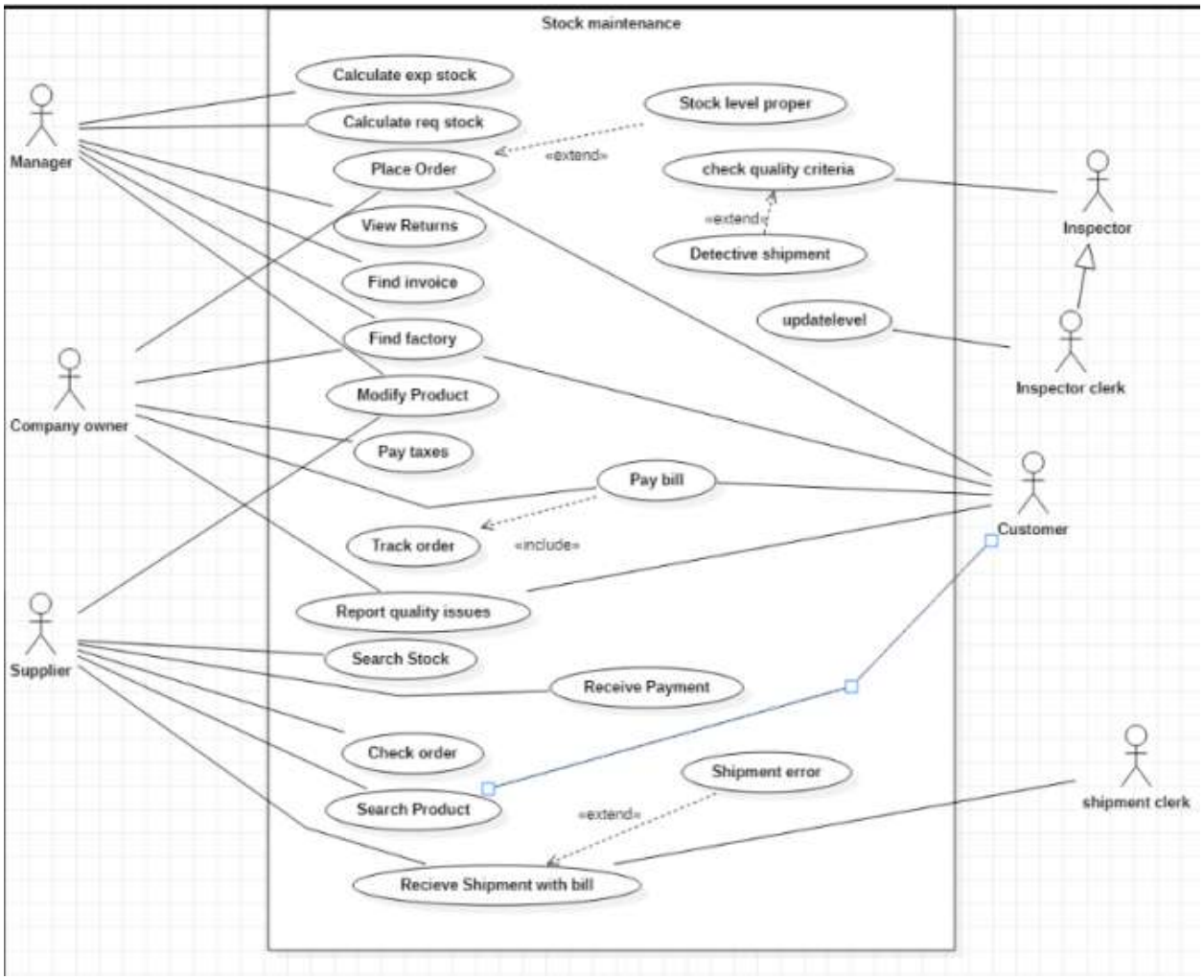


The state diagram contains one nested state and one submachine, which on successful login shows the StockStatus details and StockPurchase procedure. It contains initial state and termination state with Maintaining as a nested state including the required simple states. It also has a submachine state named StockPurchase with initial, termination state along with simple states; Inventory check, Sell, Payment, Validation.

3.4 Advanced Use case Diagram

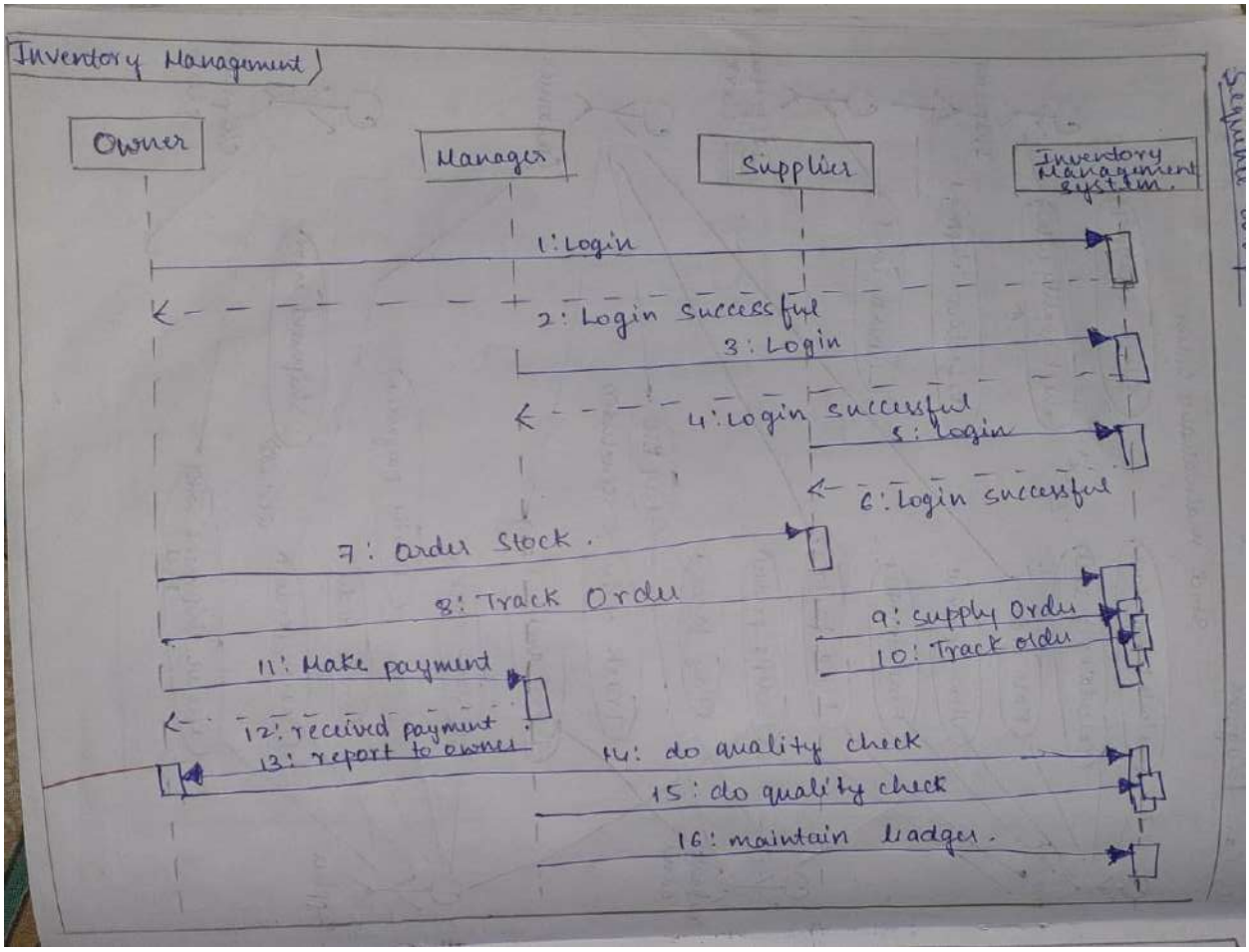
The advanced use case diagram has extra functionalities which includes extends, includes and generalization. The stock level use case extends place order use case, detective shipment use case extends check quality criteria use case , shipment error use case extends receive shipment with bill use case, pay bill use case includes track order use case.

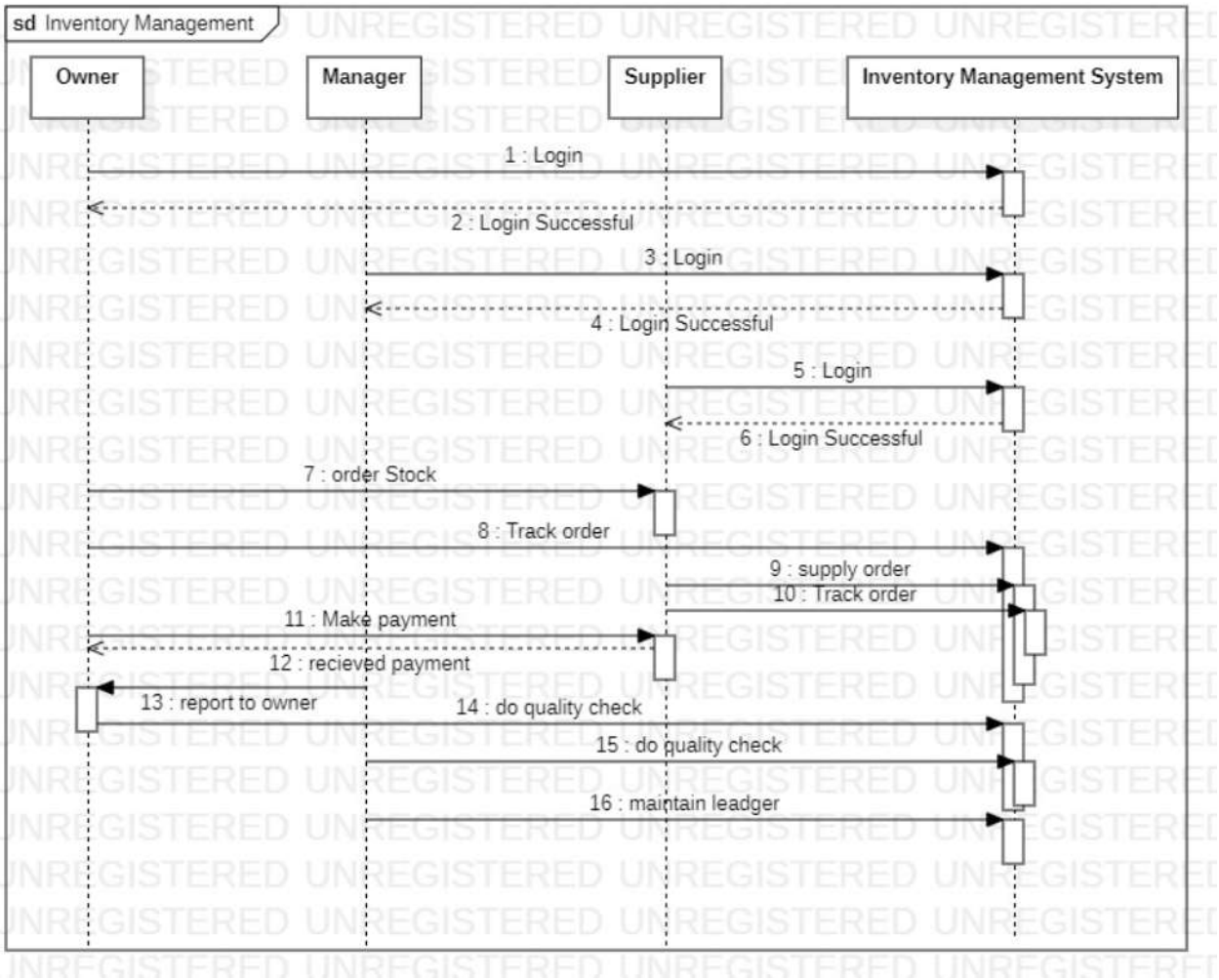

Use Case Diagram
Stock maintainance system
Calculate exp stock
Calculate req stock
Place order
View returns
Find invoice
Find factory
Modify product
Pay taxes
Pay Bill
Track order
Report quality issue
Search stock
Receive payment
Check order
Search product
Receive shipment with bill
Stock level prop
check quality situa
Defective shipment
update level
Shipment error
<<extend>>
<<include>>
<<extend>>
Manager
Company owner
Supplier
Inspector
Inspector clerk
Customer
Shipment clerk



3.5 Advanced Sequence Diagram

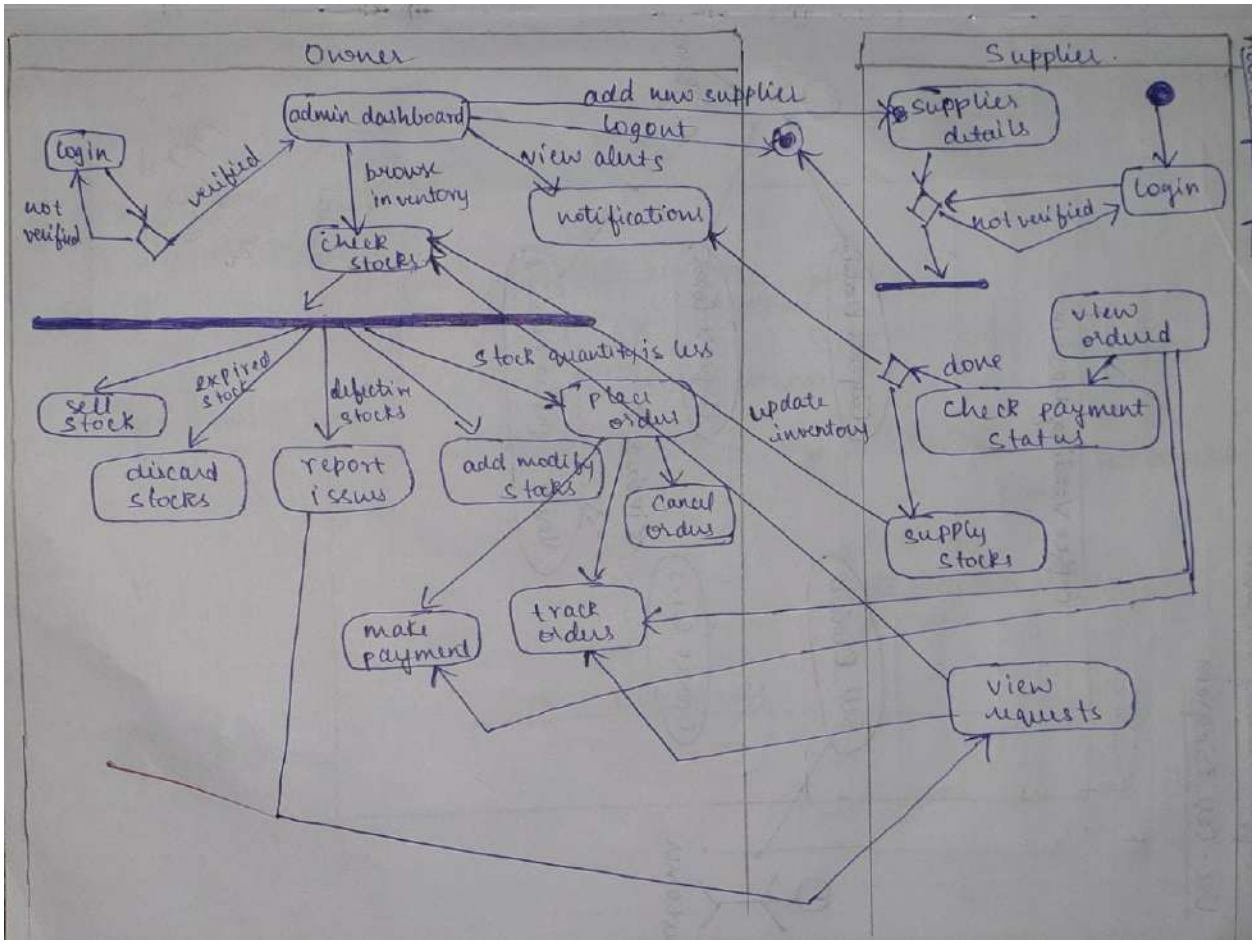
The lifeline is the dotted line and the rectangles represent the period of time the object is executing and is hence called activation. Create message signal is used to indicate the display of failure in any failure situation. The scenario shown here is the inventory management owner ordering stocks, making payment and doing quality checks on the received stocks.

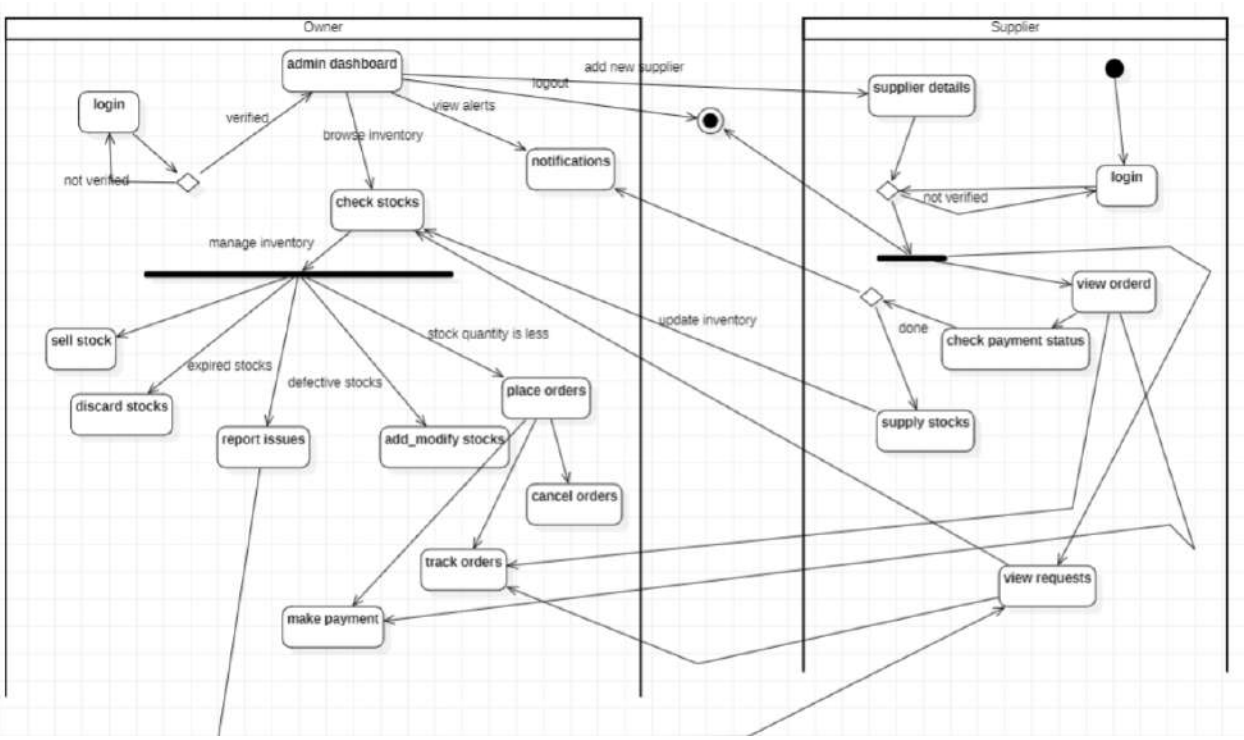

Inventory Management
Owner
Manager
Supplier
Inventory Management system.
1: Login
2: Login successful
3: Login
4: Login successful
5: Login
6: Login successful
7: Order Stock.
8: Track Order
9: supply Order
10: Track order
11: Make payment
12: received payment
13: report to owner
14: do quality check
15: do quality check
16: maintain ledger.



3.6 Advanced Activity Diagram

The advanced activity diagram starts from initiation and then user login activity where a signal is sent to the network for request validation and upon confirmation the control flows to order received and then check inventory activity. There are three swimlanes namely inventory manager, accountant and sale agent where update inventory, update payment and generate bill respectively. Then the control flows to the home page and then termination activities.


Owner
Supplier
admin dashboard
add new supplier
logout
view alerts
Login
verified
not verified
browse inventory
notifications
check stocks
supplies details
Login
not verified
view ordered
done
Check payment status
stock quantity is less
expired stock
defective stocks
sell stock
discard stocks
report issues
add modify stocks
place orders
Cancel orders
update inventory
supply stocks
make payment
track orders
view requests

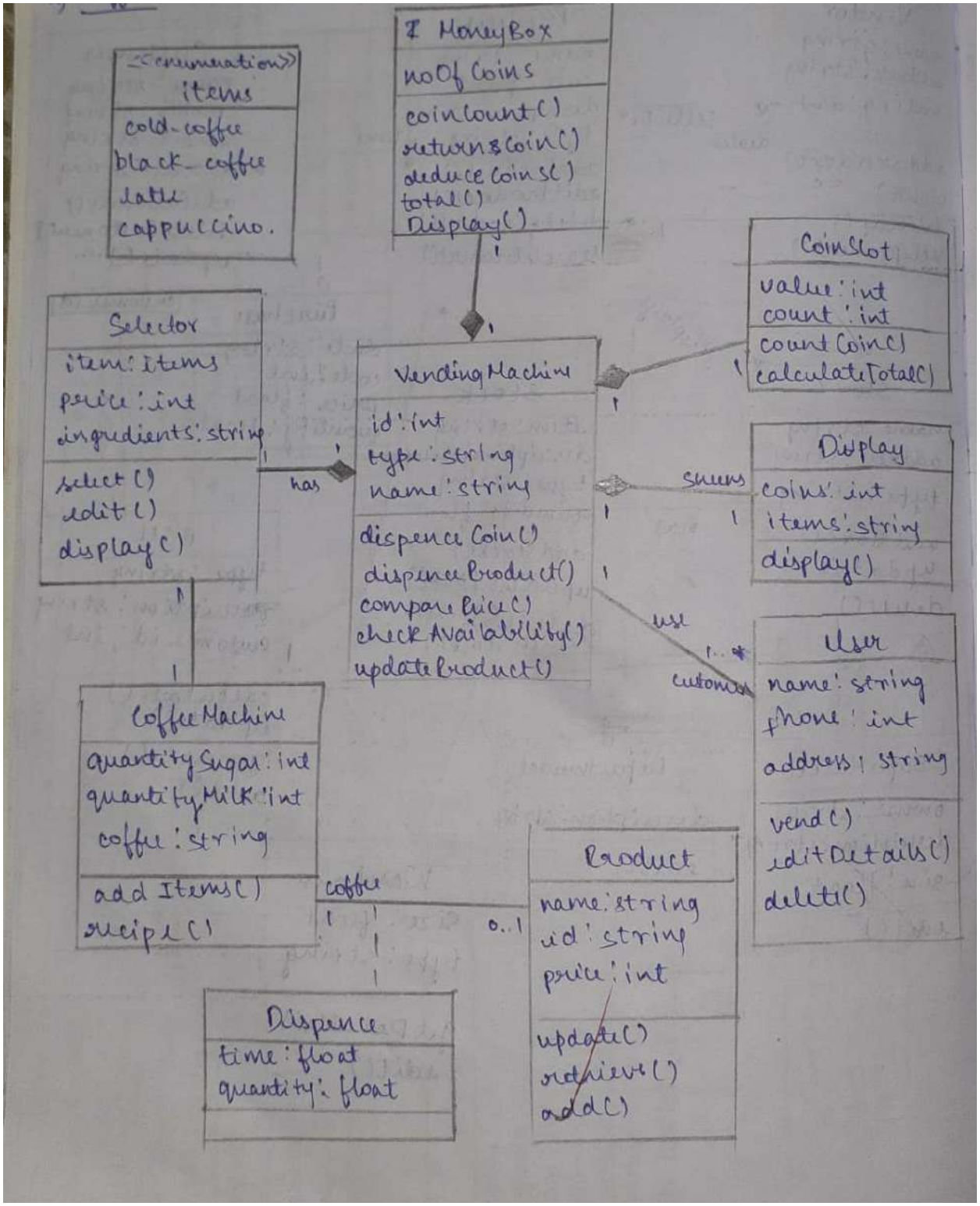

Owner
Supplier
admin dashboard
add new supplier
logout
supplier details
login
verified
view alerts
browse inventory
notifications
not verified
not verified
login
check stocks
manage inventory
view orderd
update inventory
done
check payment status
sell stock
stock quantity is less
expired stocks
defective stocks
place orders
discard stocks
supply stocks
report issues
add_modify stocks
cancel orders
track orders
view requests
make payment

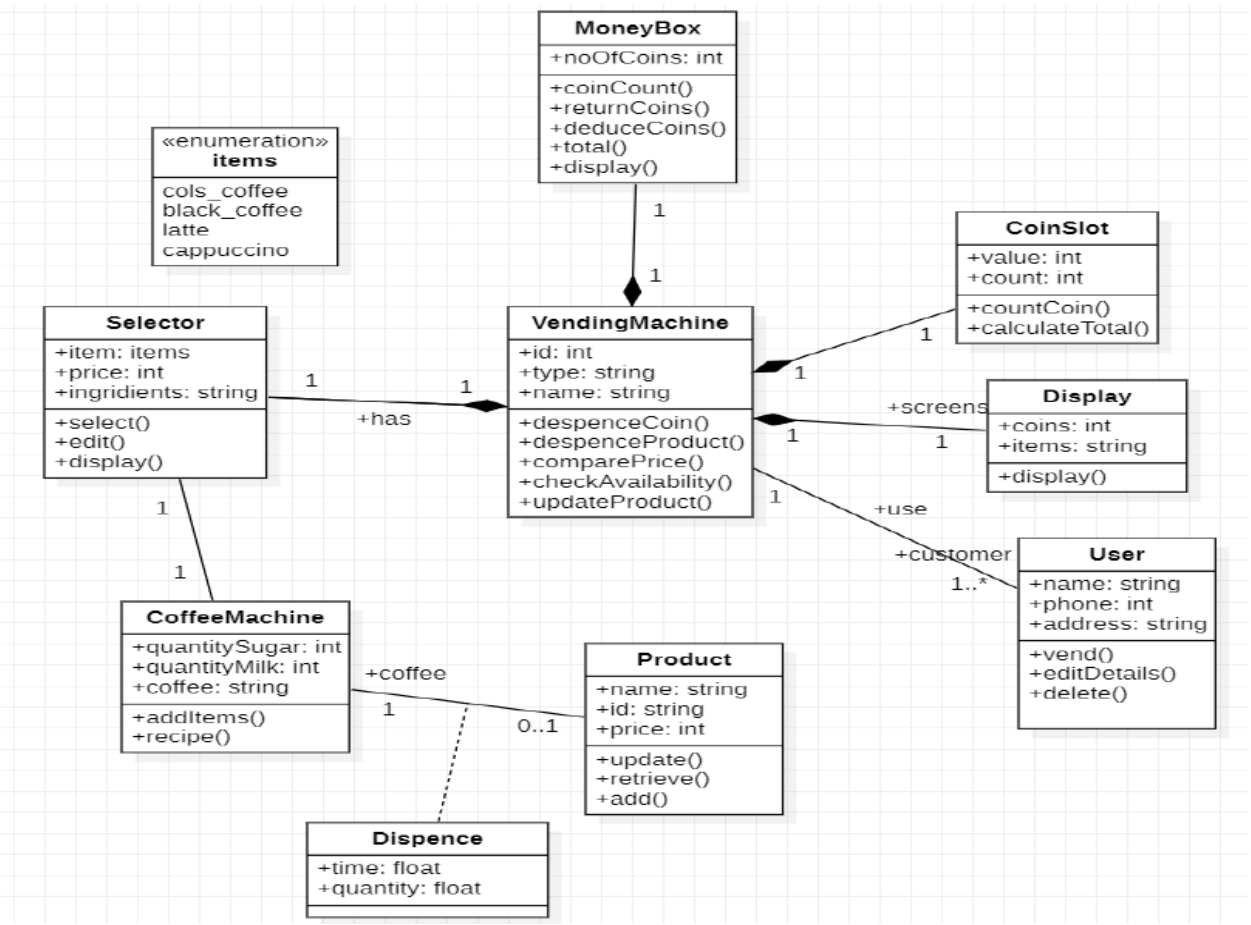
LAB 4:Coffee Vending Machine

4.1 Software Requirement Specification

- The vending machine must have a money box, coin slot, display screen and products i.e coffee for the machine to be used.
- The user on selecting a coffee ,the coffee machine must be able to dispense the selected coffee to the user.
- The user shall get an empty cup placed right below the filter.The user shall be able to choose his preferred beverage from the list of options(buttons).
- There must be buttons(start,pause,stop,coffee,tea,milk) for users 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 the dispensing.
- The system(machine) shall check for properly inserted coins.
- The system shall be able to dispense coffee(or selected beverage) after a coin has been inserted.
- The system must accept coins of different amounts and the system must compare the item cost with the 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(small LED).

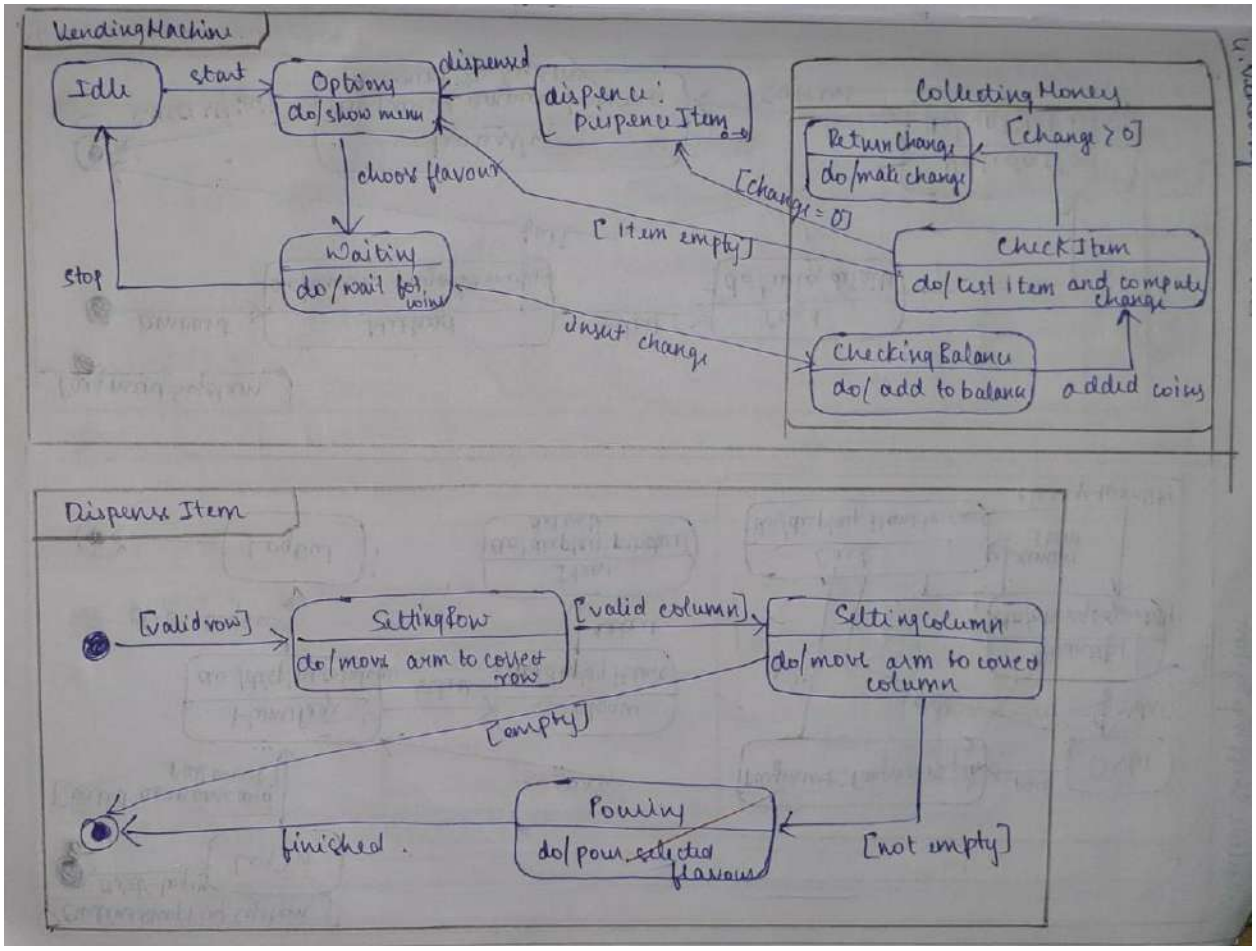
4.2 Advanced Class Diagram

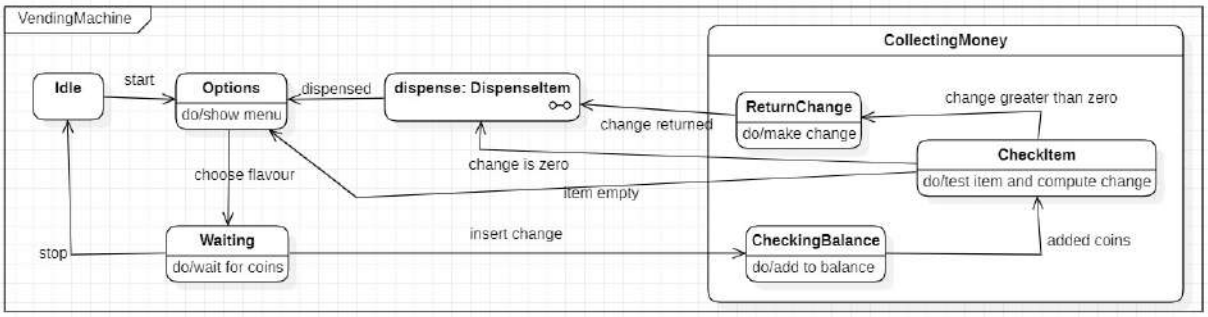

MoneyBox
noOfCoins
coinCount()
returnsCoin()
deduceCoins()
total()
Display()
<<enumeration>>
items
cold-coffe
black_coffe
latte
cappuccino.
CoinSlot
value: int
count: int
countCoin()
calculateTotal()
Selector
item: items
price: int
ingredients: string
select()
edit()
display()
has
VendingMachine
id: int
type: string
name: string
dispenceCoin()
dispenceProduct()
compareRice()
checkAvailability()
updateProduct()
shows
Display
coins: int
items: string
display()
use
customer
User
name: string
phone: int
address: string
vend()
editDetails()
delete()
CoffeeMachine
quantitySugar: int
quantityMilk: int
coffe: string
addItems()
recipe()
coffe
0..1
Product
name: string
id: string
price: int
update()
retrieve()
add()
Dispence
time: float
quantity: float

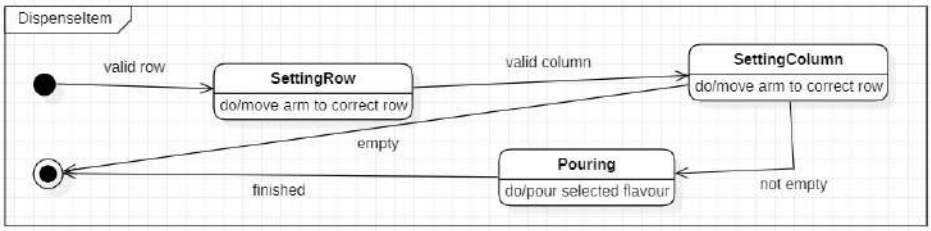


- A vending machine is a composition of a coin slot, money box, display and selector.
- The user selects the coffee through the selector. Then the coffee machine prepares the coffee.
- The coffee machine dispenses the product. Here dispense is an association class.
- Item is an enumeration of the type of coffee.

4.3 Advanced State Diagram


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

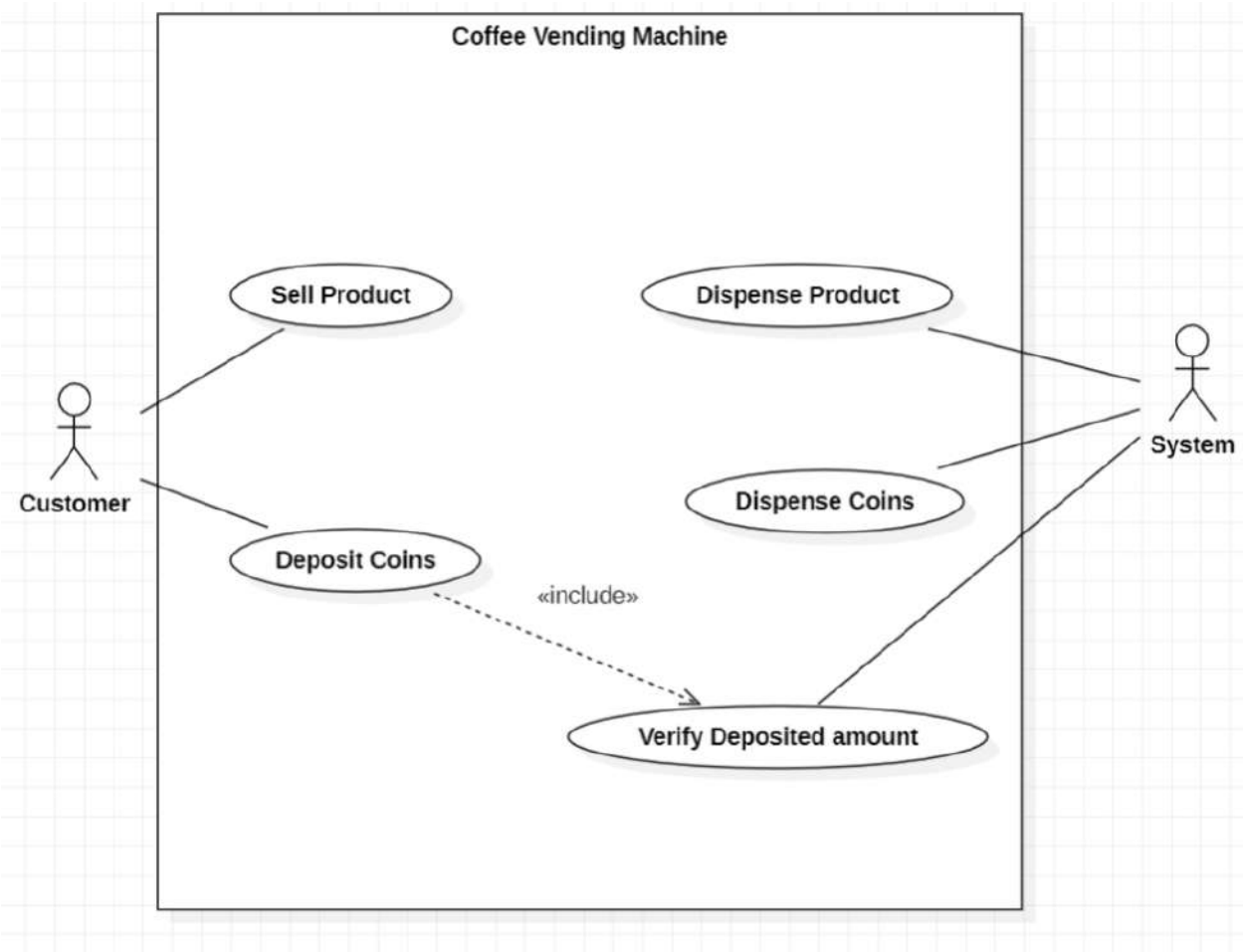

VendingMachine
Idle
start
Options
do/show menu
dispensed
dispense: DispenseItem
CollectingMoney
ReturnChange
do/make change
change returned
change greater than zero
CheckItem
do/test item and compute change
change is zero
item empty
choose flavour
Waiting
do/wait for coins
stop
insert change
CheckingBalance
do/add to balance
added coins

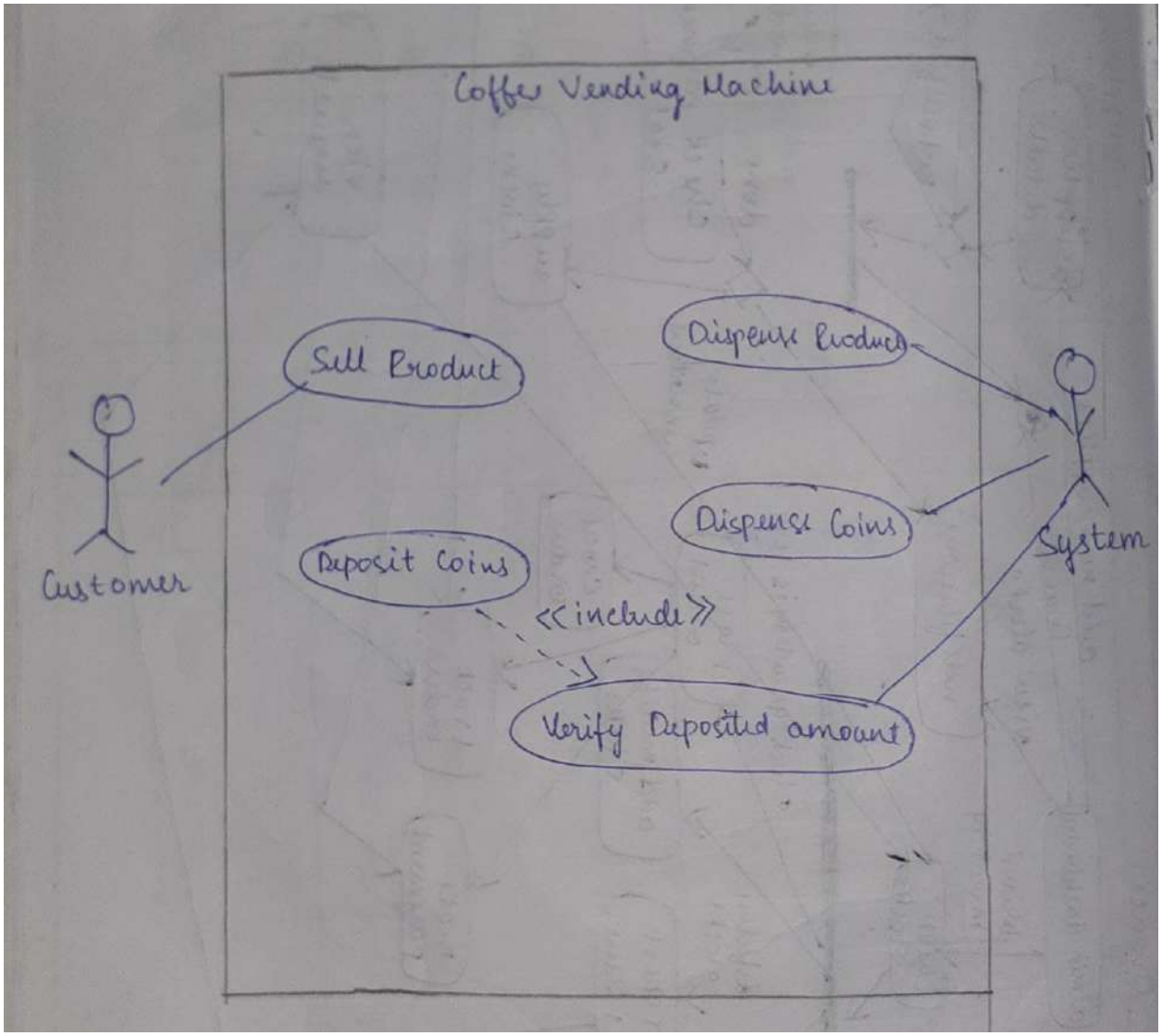

DispenseItem
valid row
SettingRow
do/move arm to correct row
valid column
SettingColumn
do/move arm to correct row
empty
Pouring
do/pour selected flavour
finished
not empty

The state diagram contains one nested state and one submachine, which on successful login shows the CollectingMoney procedure and DispenseItem procedure. It contains initial state and termination state with CollectingMoney as a nested state including the required simple states. It also has a submachine state named DispenseItem with initial, termination state along with simple states; SettingRow, SettingColumn, Pouring.

4.4 Advanced Use case Diagram

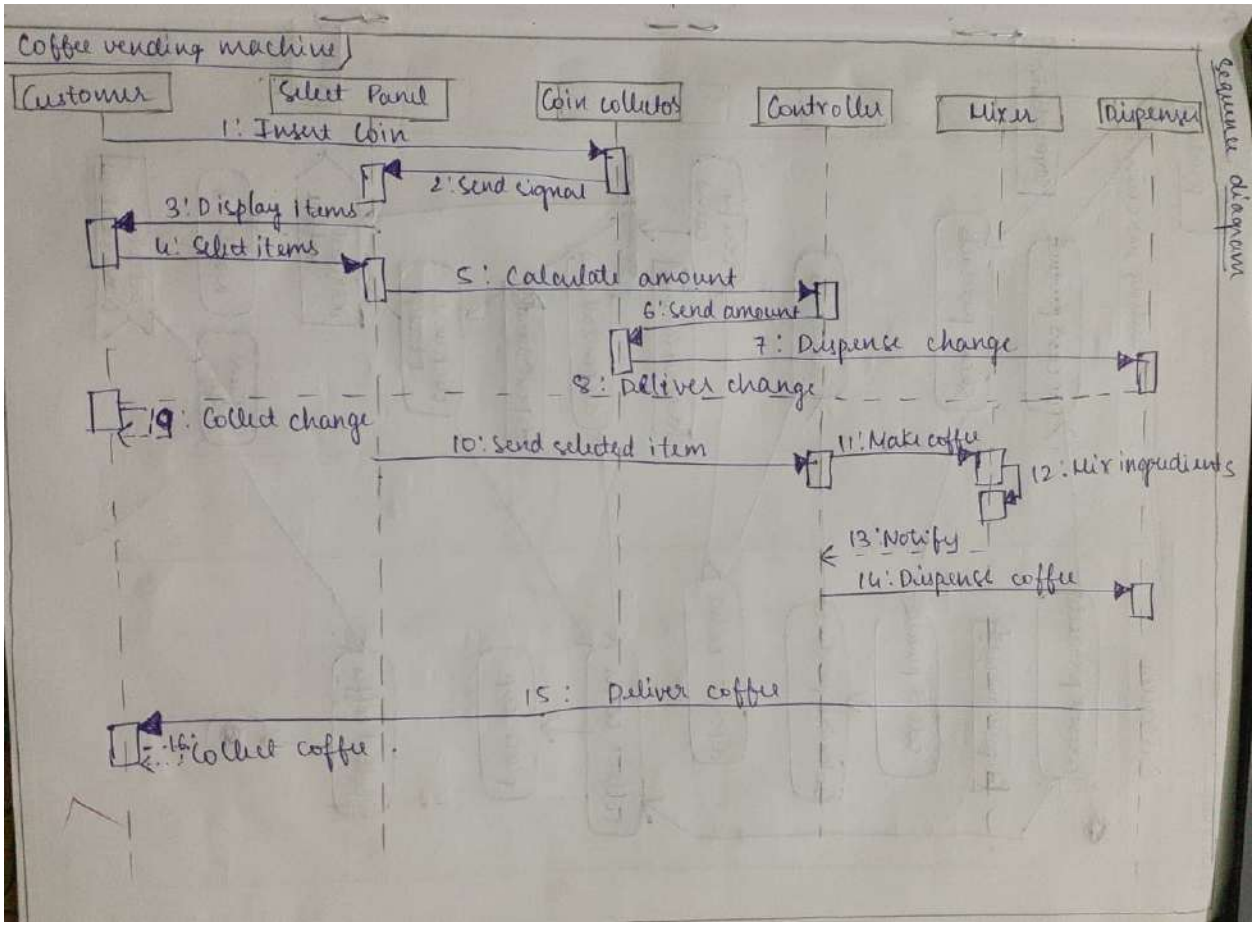
The advanced use case diagram has extra functionalities which includes extends, includes and generalization. The dispense change use case extends payment use case, payment use case extends buy item use case, buy item use case includes choose item and take item use case. Cappuccino dispenser and American dispense is generalized to super class dispense coffee

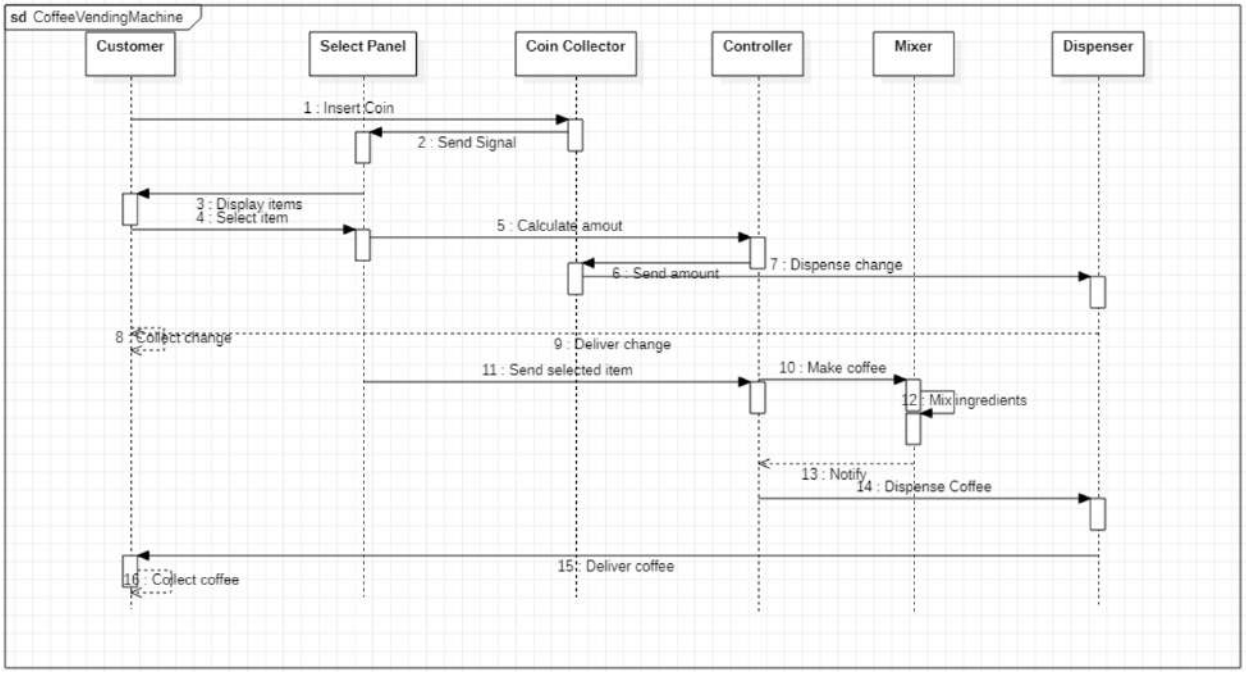




4.5 Advanced Sequence Diagram

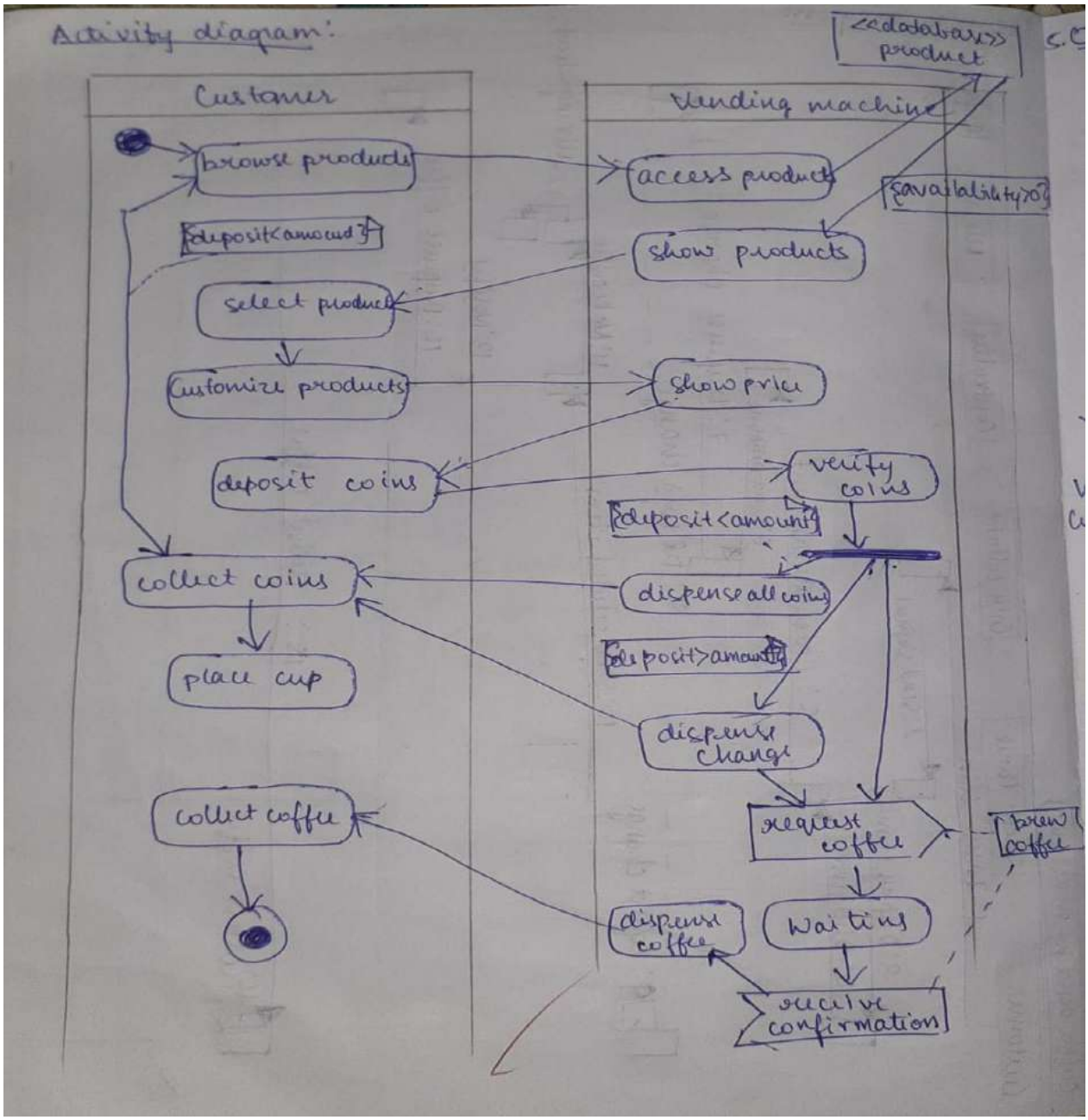
The lifeline is the dotted line and the rectangles represent the period of time the object is executing and is hence called activation. The scenario shown here is customer selecting the coffee he needs from the select panel. Then insert coins, if insufficient coins all coins are returned. Else change is dispensed and coffee is prepared and dispensed.

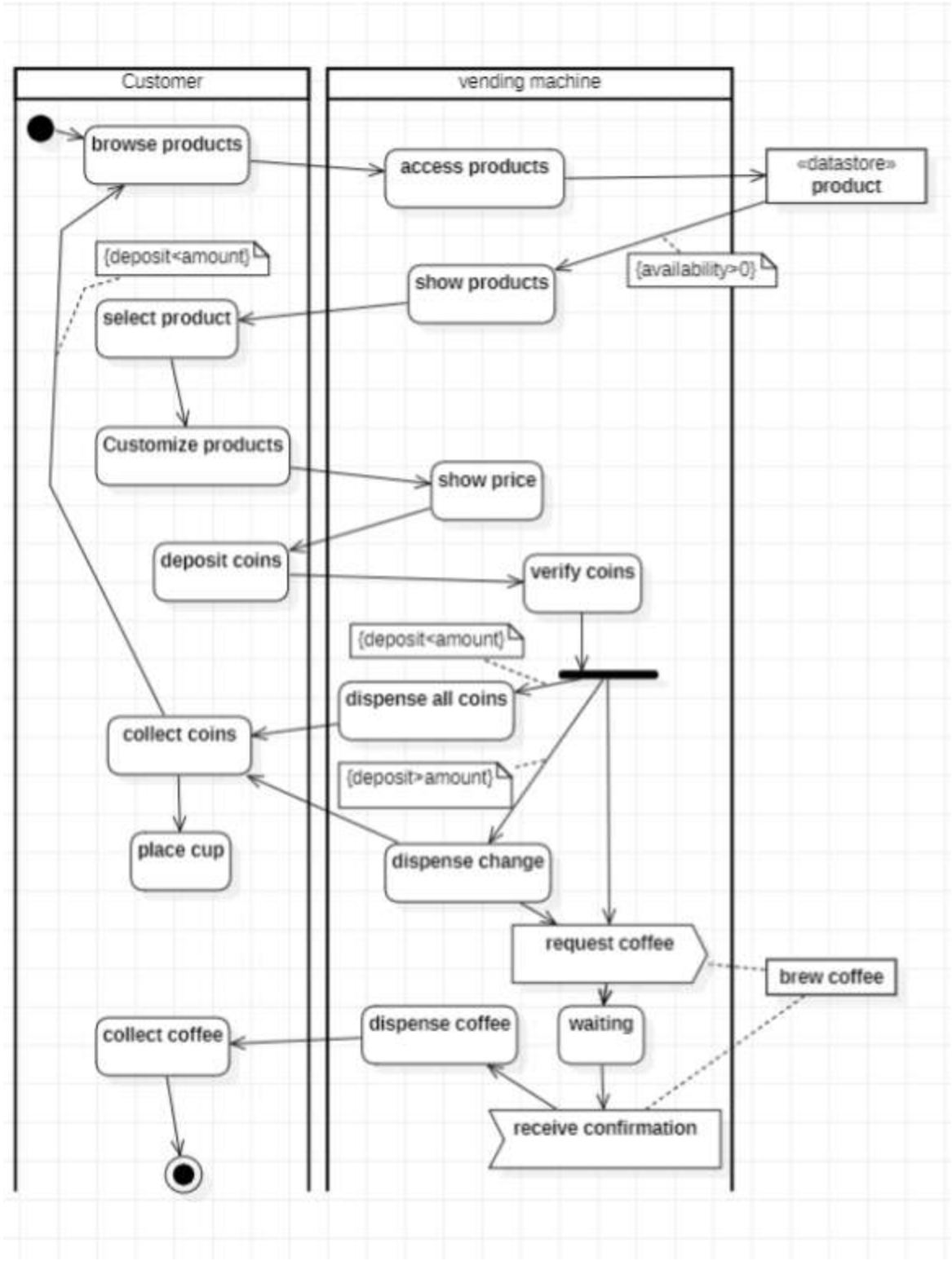




4.6 Advanced Activity Diagram

The advanced activity diagram starts from initiation and in the customer swimlane, customer browses through the products , selects the product and deposits coins. The coins deposited is verified in the vending machine swimlane and a signal is sent to brew coffee , after receiving the signal the coffee is dispensed.



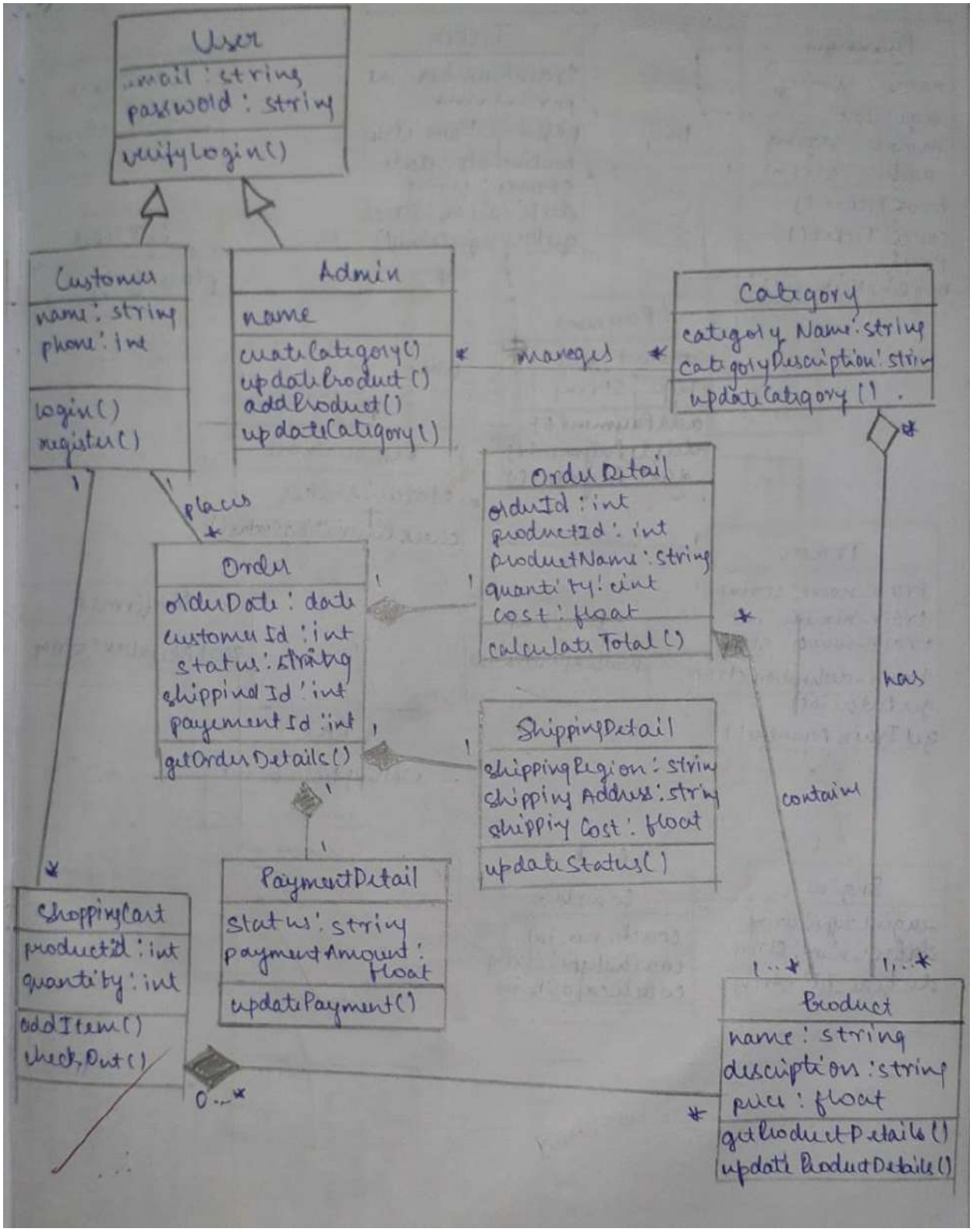

Customer
vending machine
browse products
access products
«datastore»
product
{availability>0}
{deposit<amount}
show products
select product
Customize products
show price
deposit coins
verify coins
{deposit<amount}
dispense all coins
collect coins
{deposit>amount}
place cup
dispense change
request coffee
brew coffee
collect coffee
dispense coffee
waiting
receive confirmation

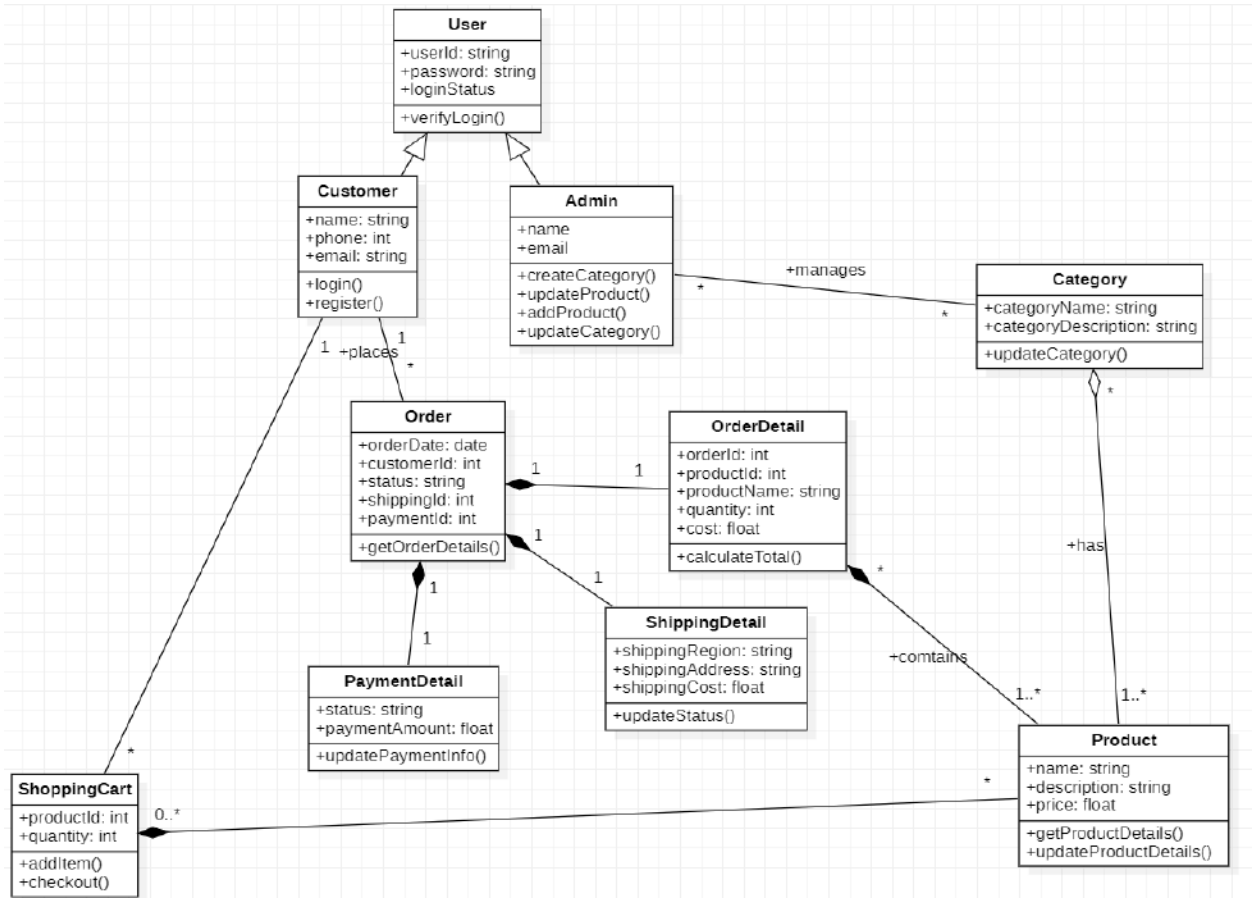
LAB 5:Online Shopping System

5.1 Software Requirement Specification

- The customer must have an account in the online website where he/she can purchase products.
- If a customer wants to buy the product then he/she must be registered, unregistered users can't go to the shopping cart.
- Customers 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 an 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.
- Customers can view all available products ,compare them and make a choice for purchasing the products.
- For customer there are many type of secure billing will be prepaid as debit or credit card, post paid as after shipping, check or bank draft. The security will provide by the third party like Pay-Pal etc.
- After the payment or surf the product the customer will logged out.

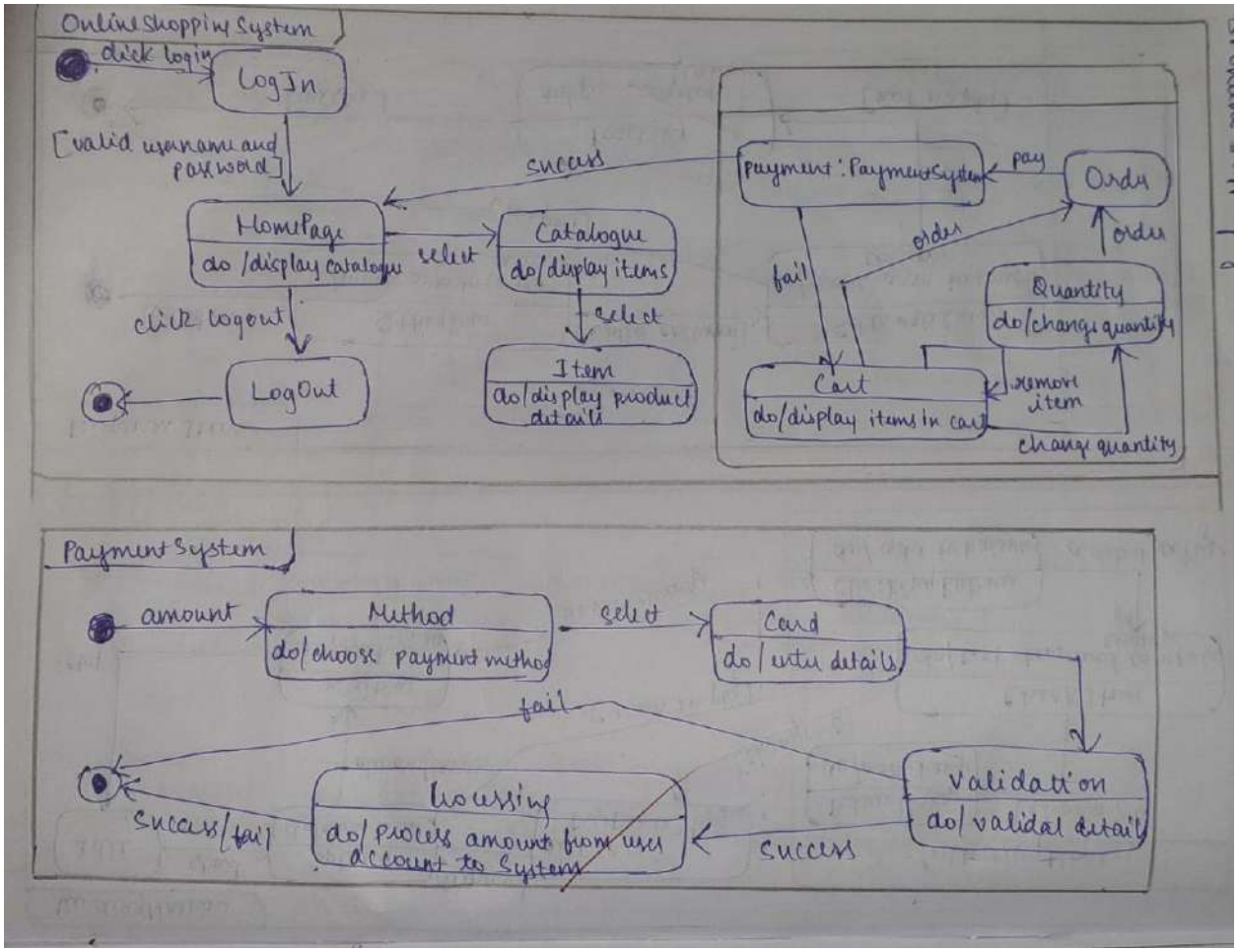
5.2 Advanced Class diagram

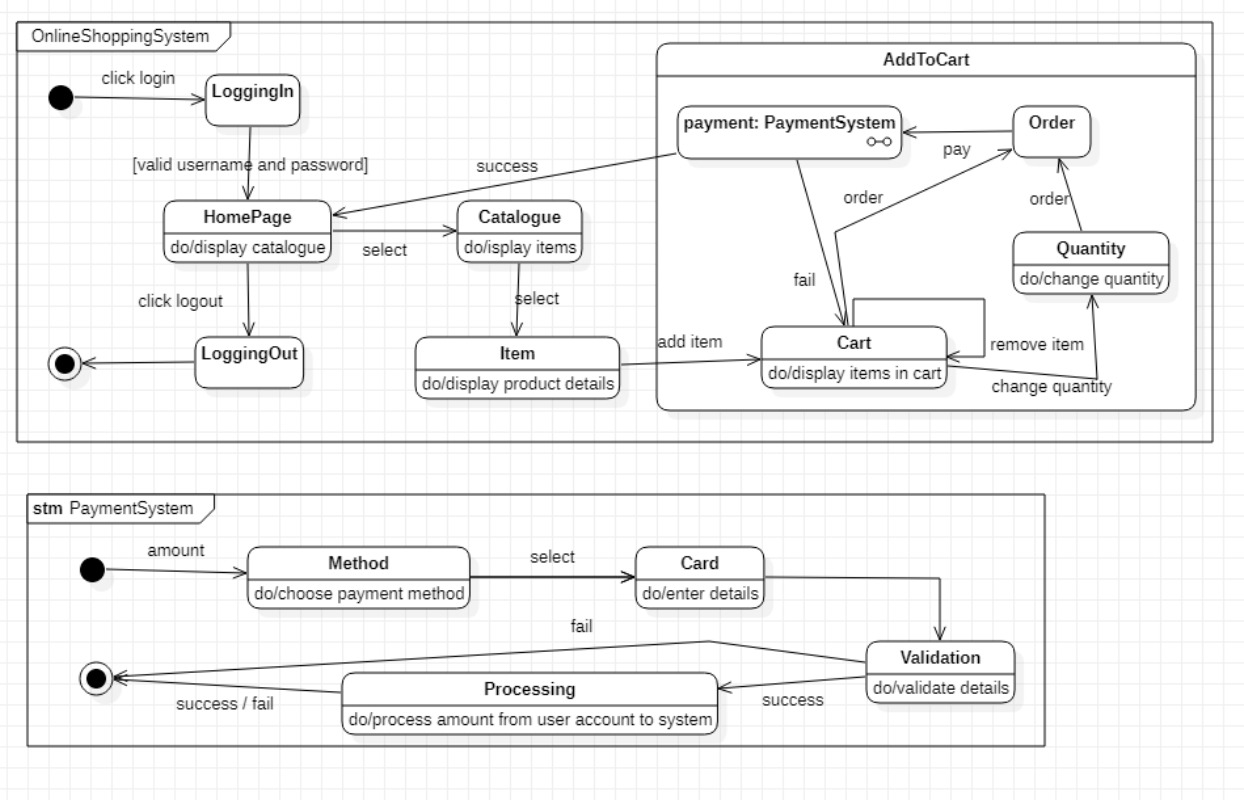

User
email : string
password : string
verifyLogin()
Customer
name : string
phone : int
login()
register()
Admin
name
createCategory()
updateProduct()
addProduct()
updateCategory()
manages
Category
categoryName : string
categoryDescription : string
updateCategory()
places
Order
orderDate : date
customerId : int
status : string
shippingId : int
paymentId : int
getOrderDetails()
OrderDetail
orderId : int
productId : int
productName : string
quantity : int
cost : float
calculateTotal()
has
ShippingDetail
shippingRegion : string
shippingAddress : string
shippingCost : float
updateStatus()
contains
PaymentDetail
status : string
paymentAmount : float
updatePayment()
ShoppingCart
productId : int
quantity : int
addItem()
checkOut()
Product
name : string
description : string
price : float
getProductDetails()
updateProductDetails()



- An online shopping system user can either be a customer or an admin. So the user is generalized into customer and admin.
- Admin manages the category of different products. He is responsible for adding products.
- The user places orders. An order is a composition of Order detail, shipping detail and payment detail.
- Customer can also add 0 or more product to the shopping cart

5.3 Advanced State Diagram

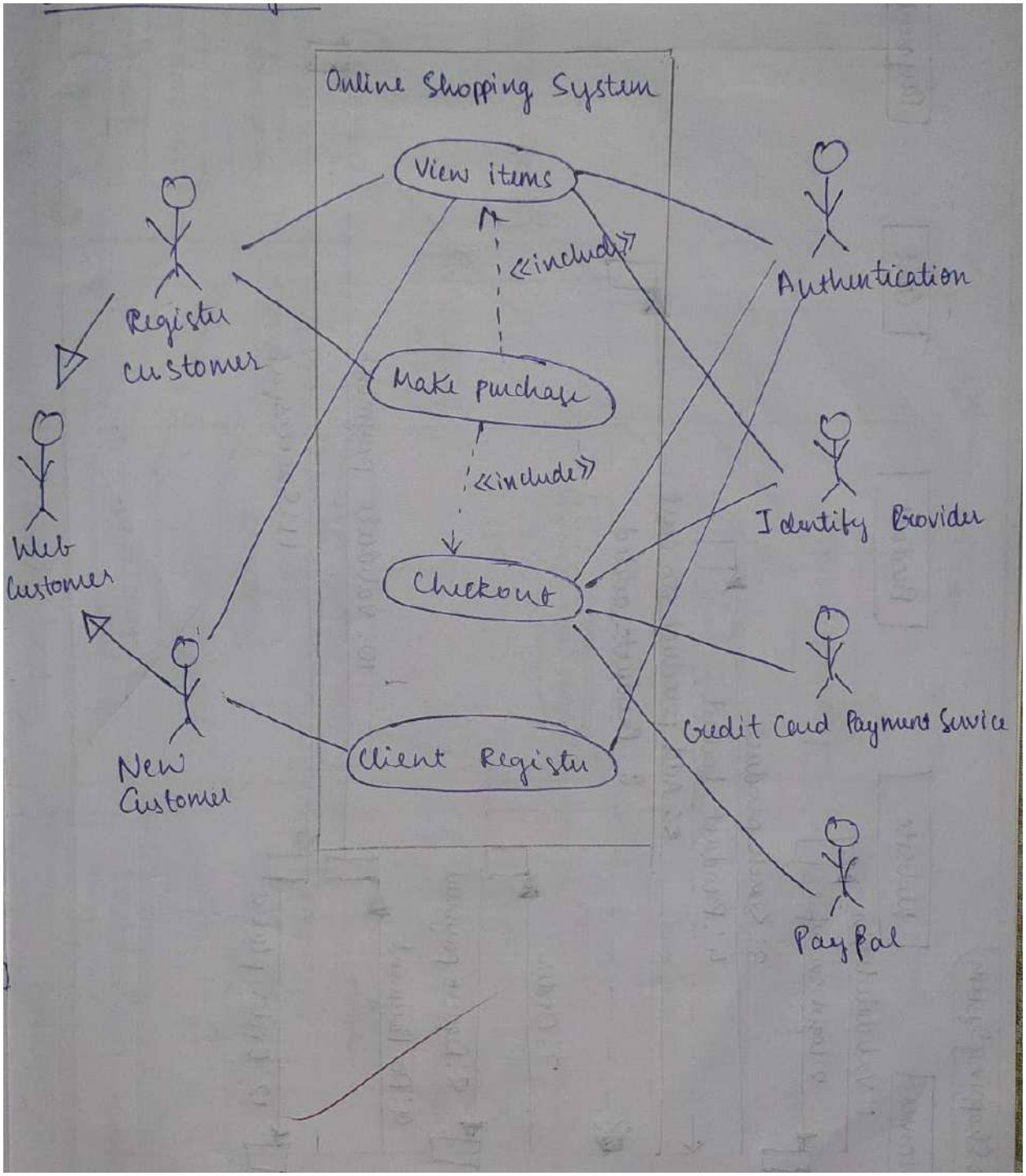

Online Shopping System
click login
LogIn
[valid username and password]
HomePage
do/display catalogue
select
Catalogue
do/display items
select
Item
do/display product details
click logout
LogOut
success
Payment : PaymentSystem
pay
Order
order
order
fail
Quantity
do/change quantity
Cart
do/display items in cart
remove item
change quantity
Payment System
amount
Method
do/choose payment method
select
Card
do/enter details
fail
Validation
do/validate details
success
Processing
do/process amount from user account to System
success/fail

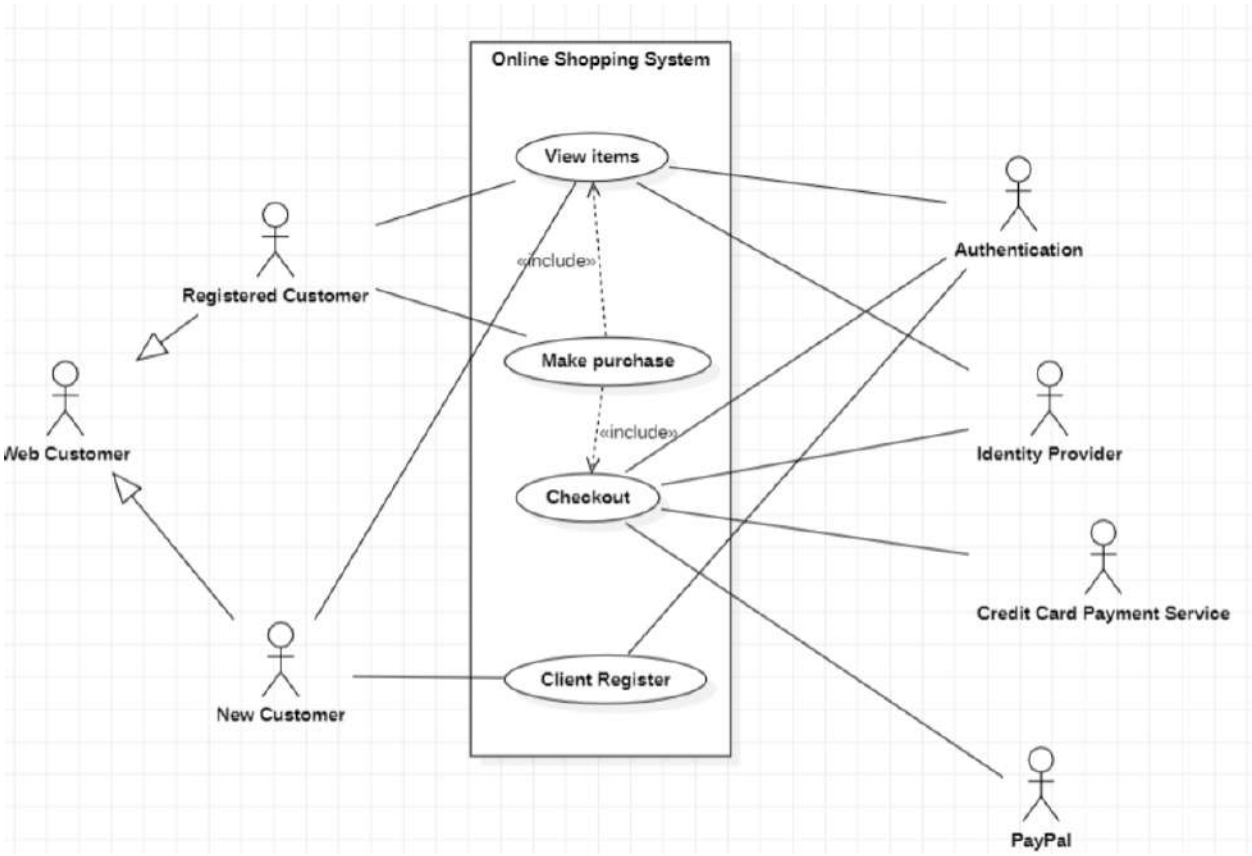


The state diagram contains one nested state and one submachine, which on successful login shows the AddToCart procedure and PaymentSystem procedure. It contains initial state and termination state with AddToCart as a nested state including the required simple states. It also has a submachine state named PaymentSystem with initial, termination state along with simple states; Method, Card, Validation, Processing.

5.4 Advanced use case diagram

The advanced use case diagram has extra functionalities which includes extends, includes and generalization. Web customer is generalized into registered customer and new customer. Only registered customers can make purchases, while new customers can still view items.

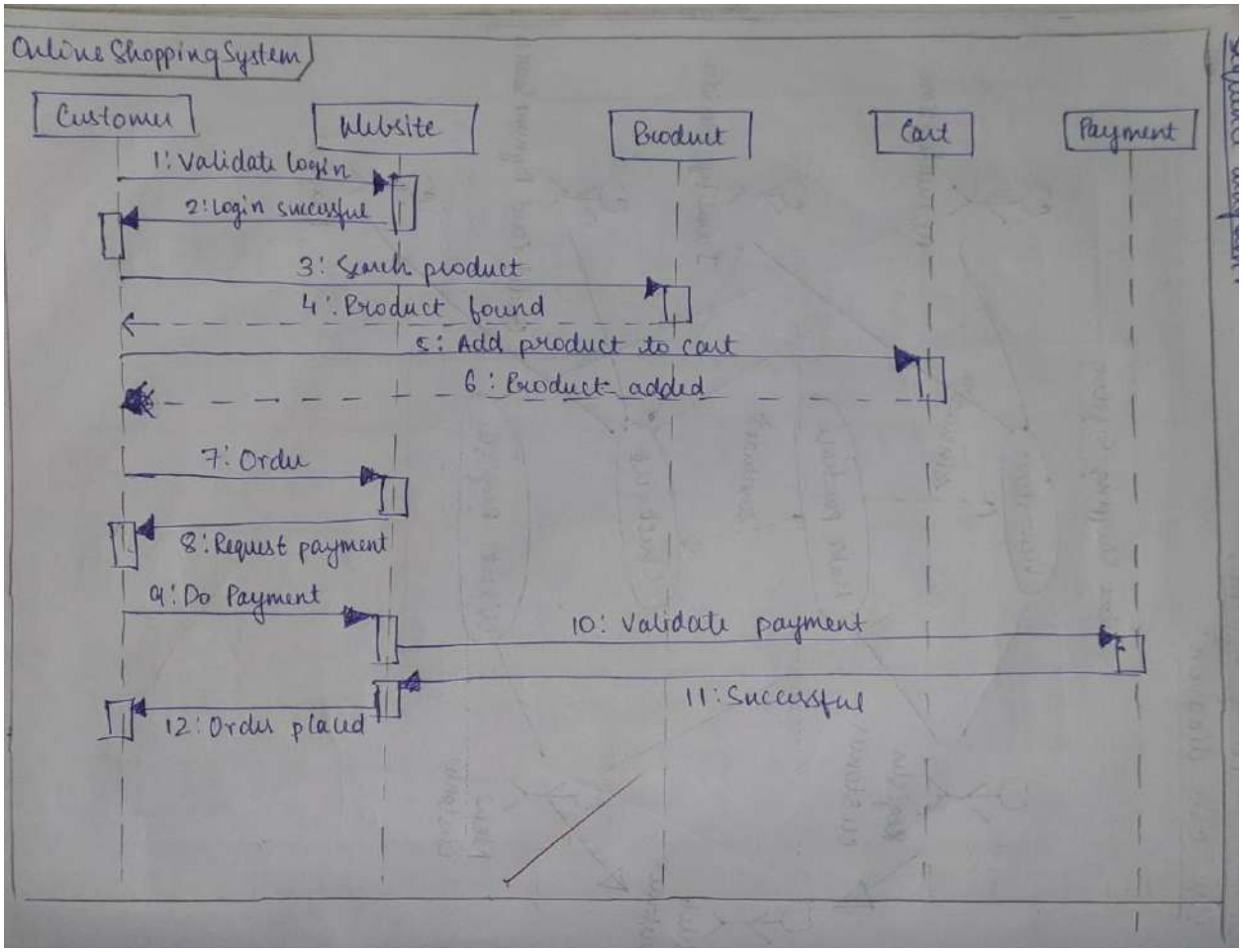

Online Shopping System
View items
«include»
Make purchase
«include»
Checkout
Client Registry
Register customer
Web Customer
New Customer
Authentication
Identity Provider
Credit Card Payment Service
PayPal

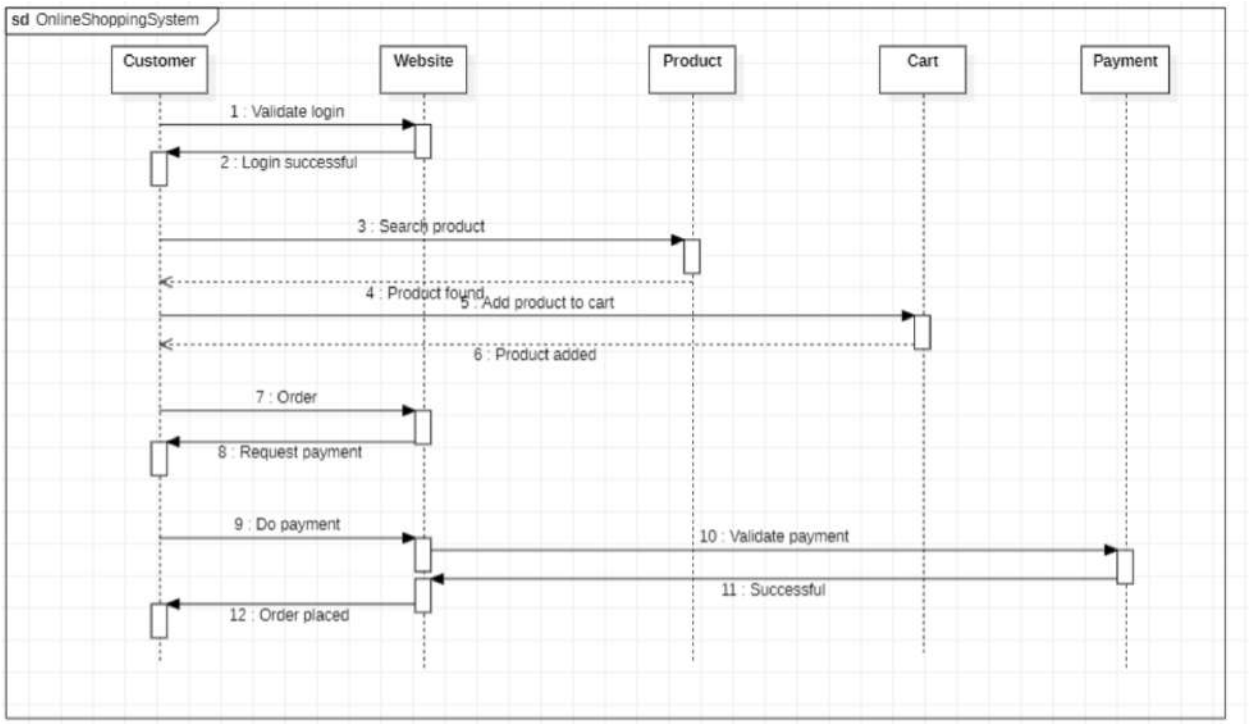


5.5 Advanced Sequence diagram

The actors are: Customer website product cart payment

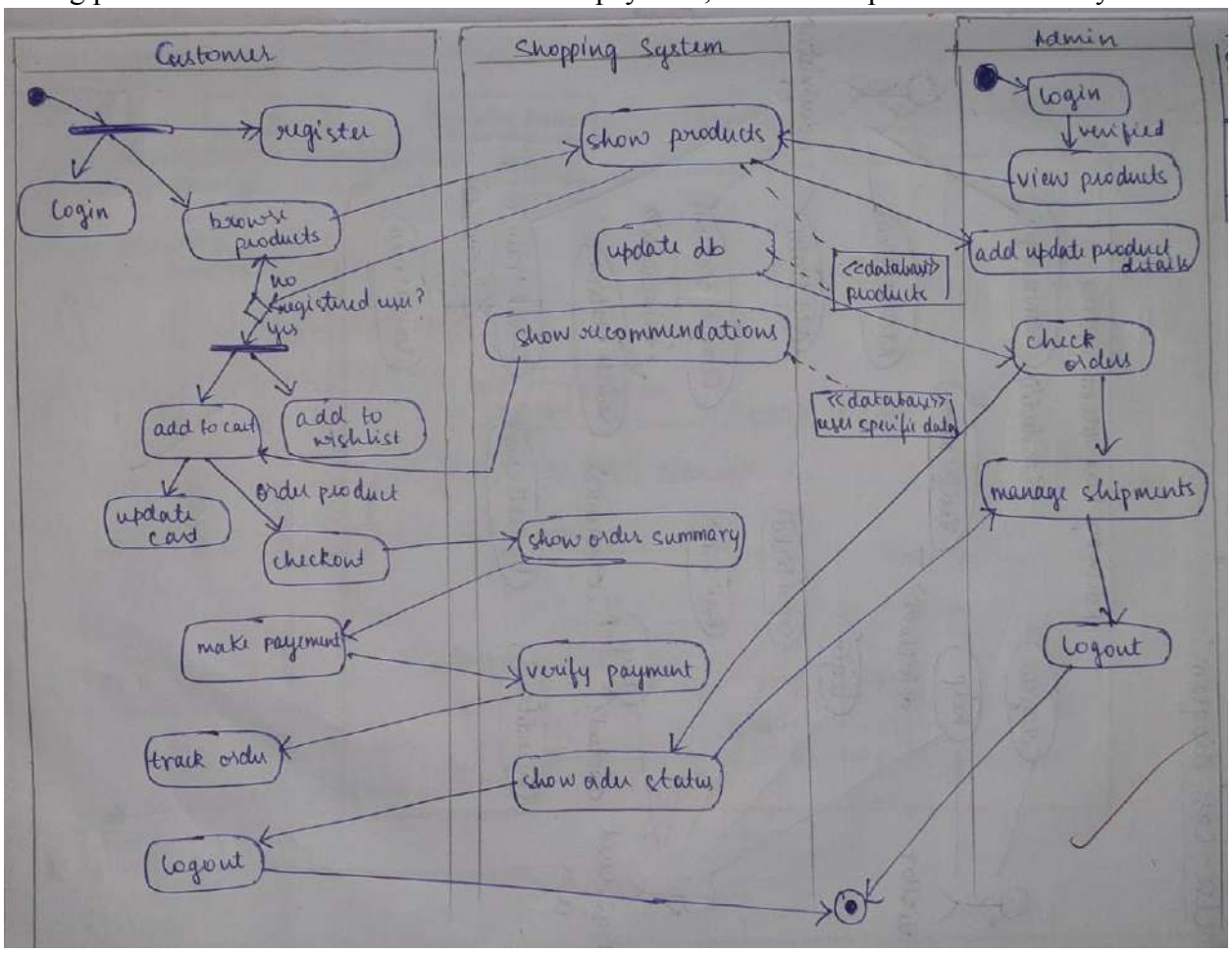
The scenario shown here is customer logging in into the website, searching for the product then adding product to cart checkout. On successful payment, the order is placed successfully.

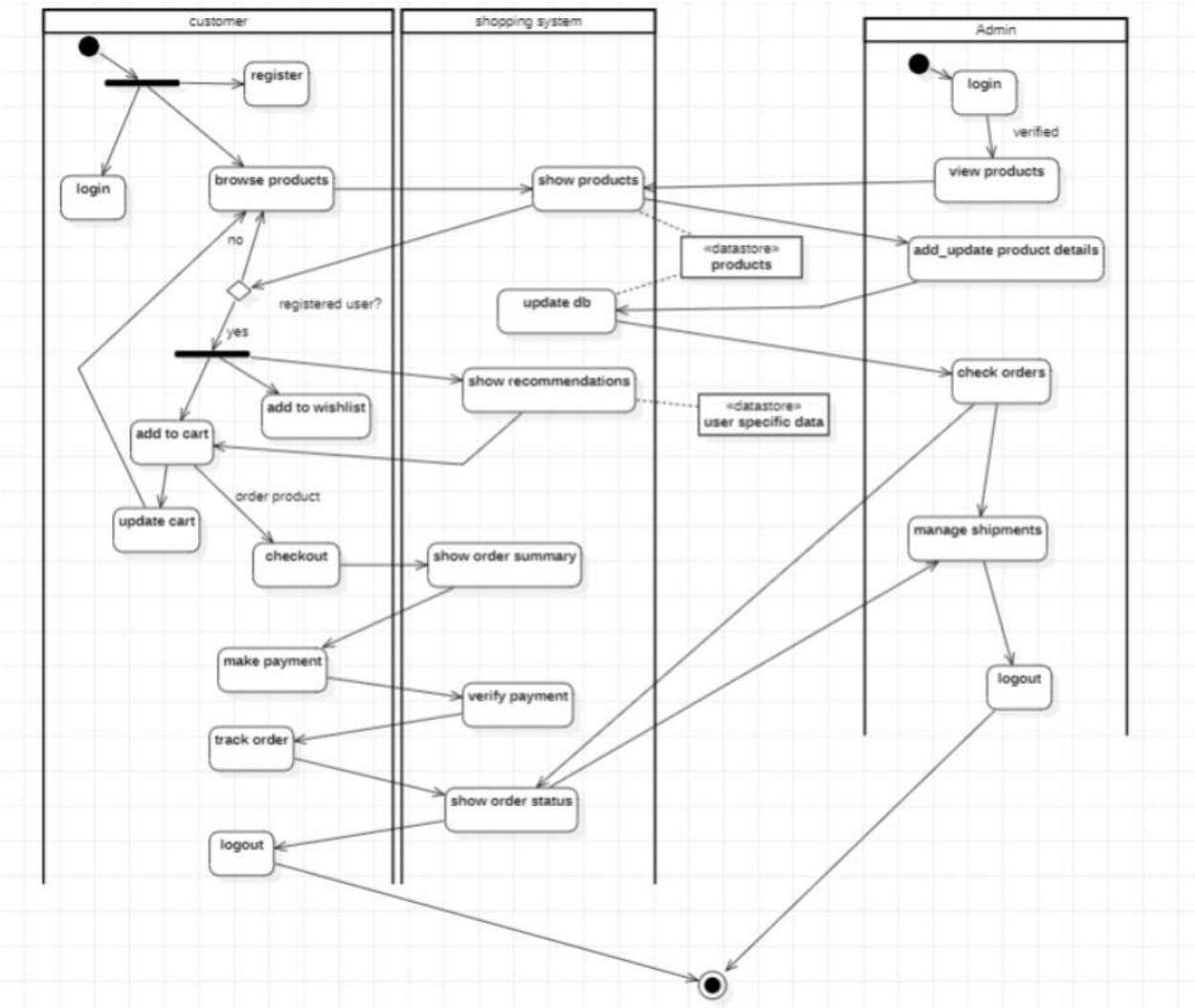

Online Shopping System
Customer
Website
Product
Cart
Payment
1: Validate login
2: Login successful
3: Search product
4: Product found
5: Add product to cart
6: Product added
7: Order
8: Request payment
9: Do Payment
10: Validate payment
11: Successful
12: Order placed


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

5.6 Advanced Activity diagram

There are 3 swimlanes: Customer, shopping system and admin.
The scenario shown here is customer logging in into the website, searching for the product then adding product to cart checkout. On successful payment, the order is placed successfully.



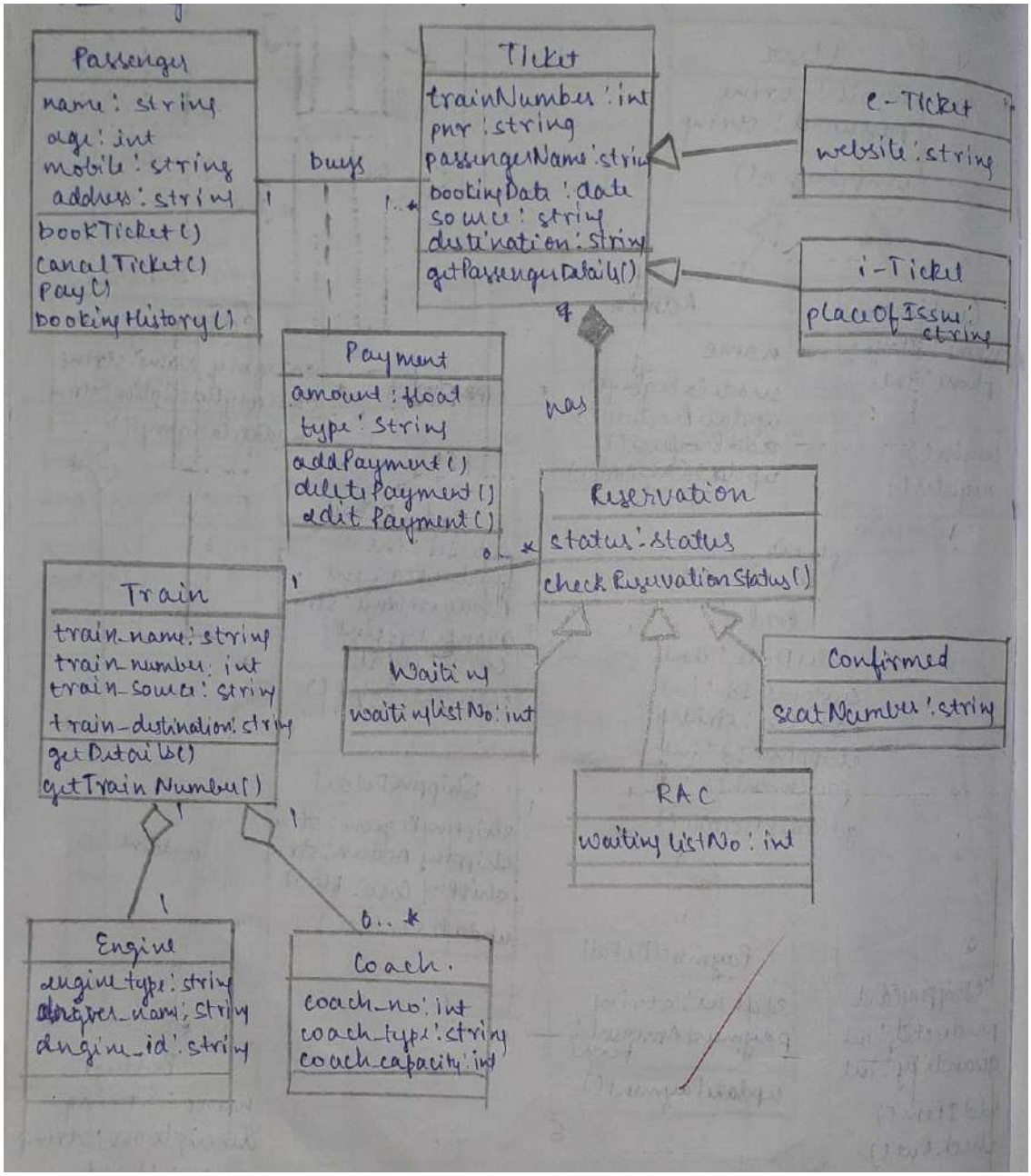

customer
shopping system
Admin
register
login
browse products
show products
login
verified
view products
no
«datastore»
products
add_update product details
registered user?
update db
yes
show recommendations
check orders
add to wishlist
«datastore»
user specific data
add to cart
order product
update cart
checkout
show order summary
manage shipments
make payment
verify payment
logout
track order
show order status
logout

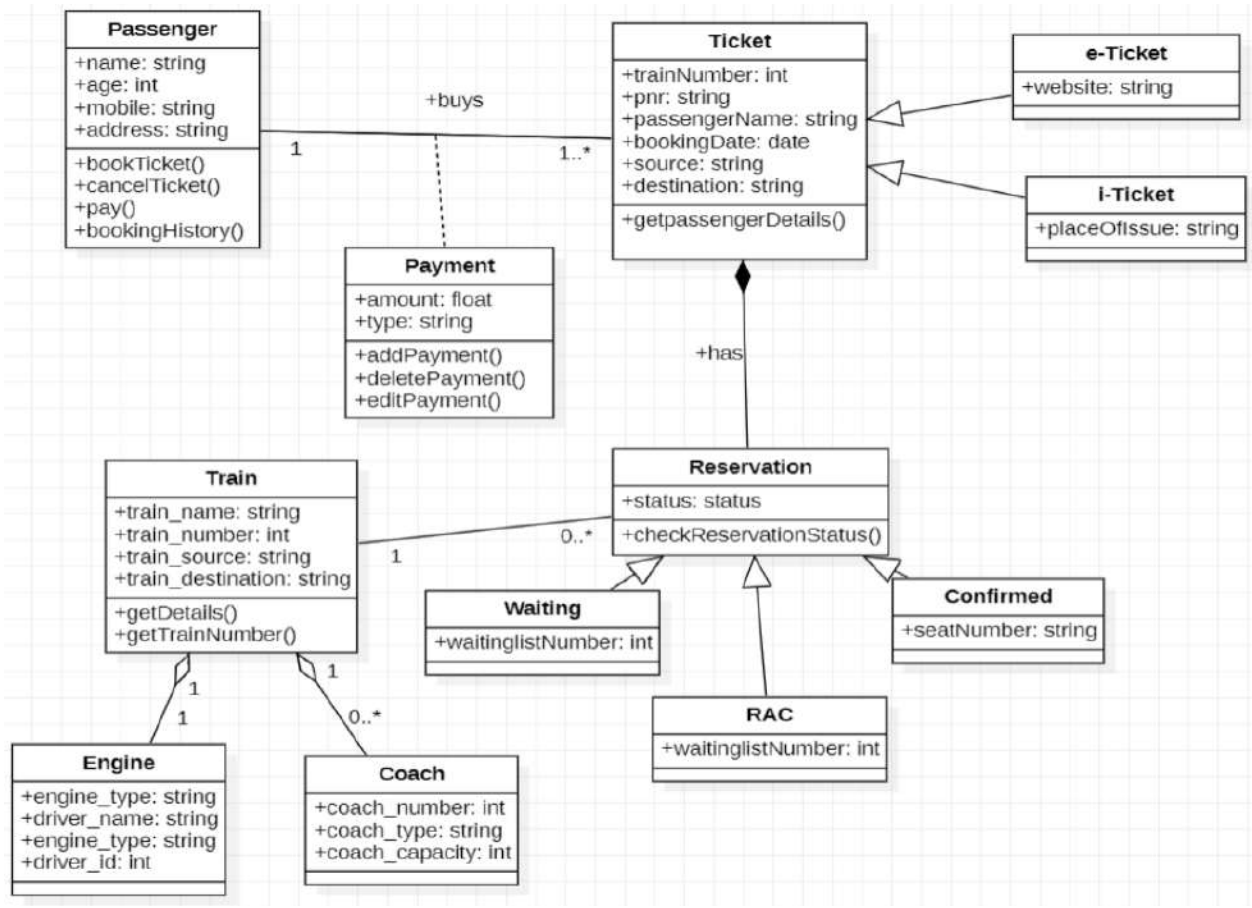
LAB 6:Railway Reservation Machine

6.1 Software Requirement 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 the tickets by initially filling details in a form.
- Tickets can be booked in two ways by i-ticket or by e-ticket booking.
- 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-ticket booking and canceling tickets are booked and canceled online sitting at the home and customer himself has to take print of the ticket but in both the cases amount for tickets are deducted from customers account.
- For cancellation of a ticket the customer has to go to the reservation office then fill in the cancellation form and ask the clerk to cancel the ticket then the refund is transferred to the customer account.
- After booking a ticket the customer has to checkout by paying fare amount to the clerk.
- The system displays the details of the train of which the user enters the name.The Information is saved and the corresponding updating takes place in the database.

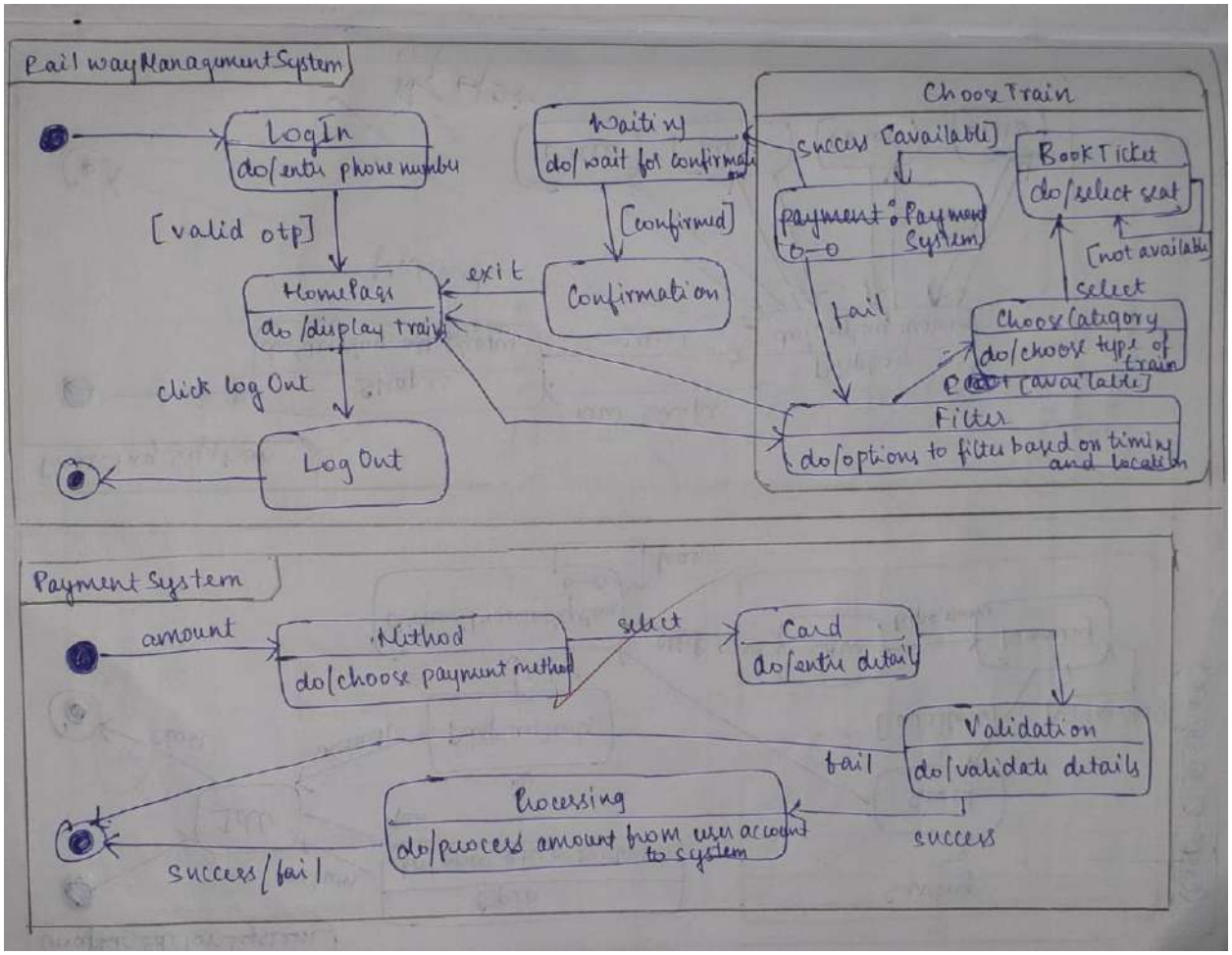
6.2 Advanced Class Diagram

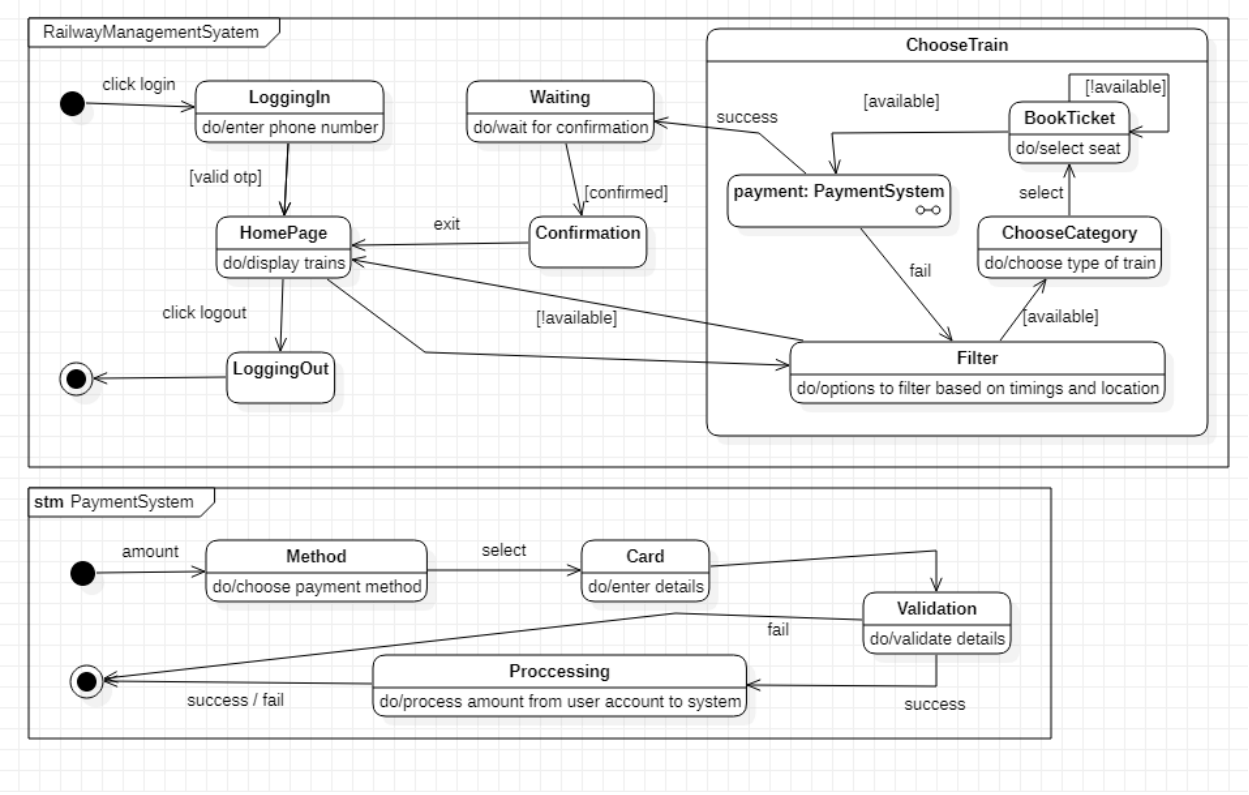

Passenger
name : string
age : int
mobile : string
address : string
bookTicket()
cancelTicket()
Pay()
BookingHistory()
buys
1
1..*
Ticket
trainNumber : int
pnr : string
passengerName : string
bookingDate : date
source : string
destination : string
getPassengerDetails()
e-Ticket
website : string
i-Ticket
placeOfIssue : string
has
1
Payment
amount : float
type : String
addPayment()
deletePayment()
editPayment()
0..*
Reservation
status : status
checkReservationStatus()
Train
train_name : string
train_number : int
train_source : string
train_destination : string
getDetails()
getTrainNumber()
1
1
Waiting
waitinglistNo : int
Confirmed
seatNumber : string
RAC
waitingListNo : int
1
1
1
0..*
Engine
engine_type : string
engine_name : string
engine_id : string
Coach
coach_no : int
coach_type : string
coach_capacity : int



- Passenger books ticket. The ticket is generalized as i-ticket and e-ticket.
- Reservation cannot exist without a ticket. So the ticket has a composition relation with the ticket.
- Reservation can be generalized as waiting, RAC and confirmed based on the status of the ticket.
- Train is made of 1 engine and 0 or more coaches. The coaches and engine can exist without the train,so aggregation is used here

6.3 Advanced State Diagram


Rail way Managment System
Log In
do/ entir phone numbr
[valid otp]
Waiting
do/ wait for confirmati
[confirmed]
exit
Confirmation
Home Page
do /display train
click log Out
Log Out
Choose Train
Succes [available]
payment
Payment System
fail
Book Ticket
do/select seat
[not available]
select
Choose Catigory
do/choose type of train
[available]
Filter
do/options to filter based on timing and location
Payment System
amount
Method
do/choose payment method
select
Card
do/entir detail
Validation
do/validate details
fail
success
Processing
do/process amount from user account to system
success/fail

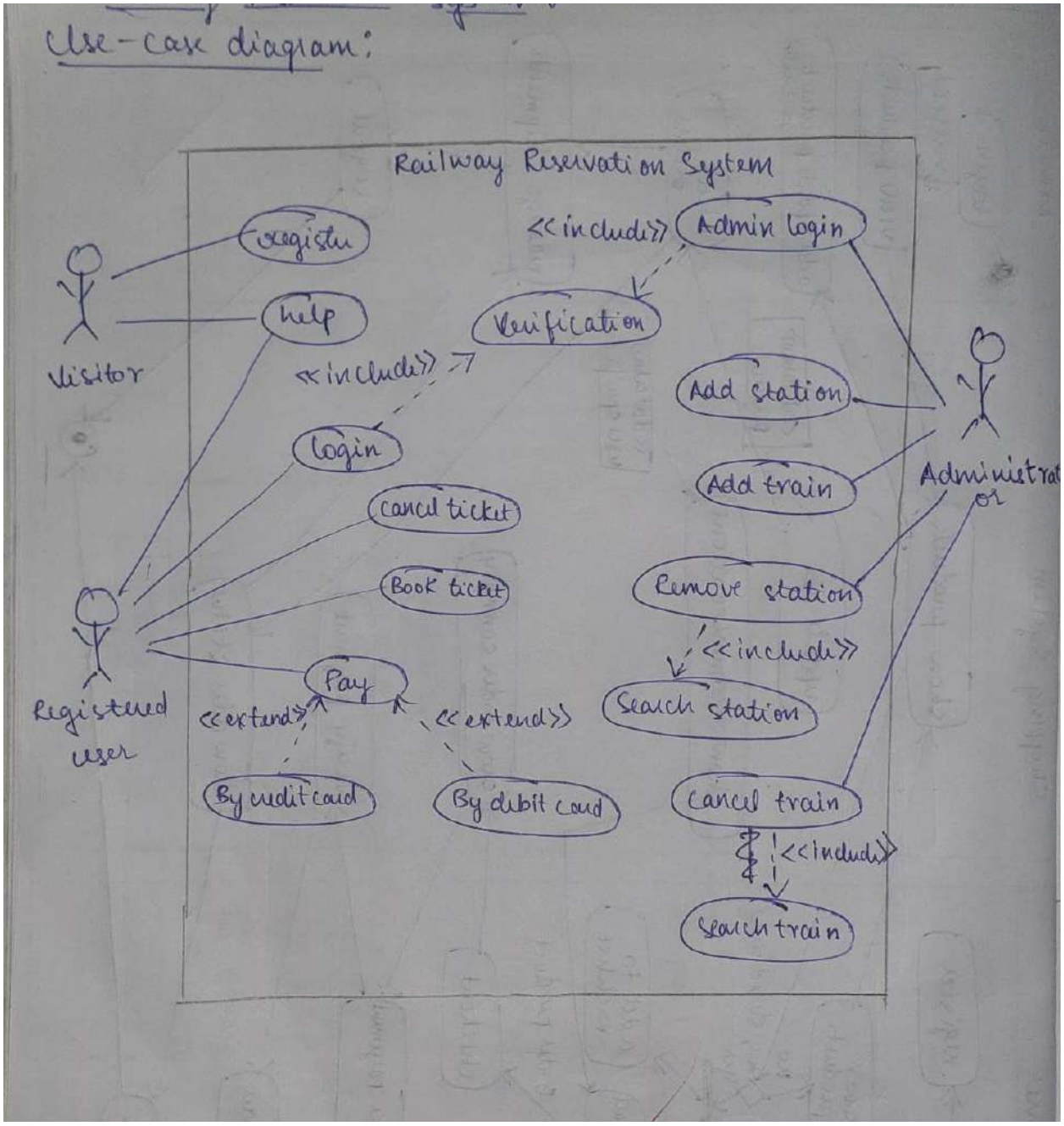


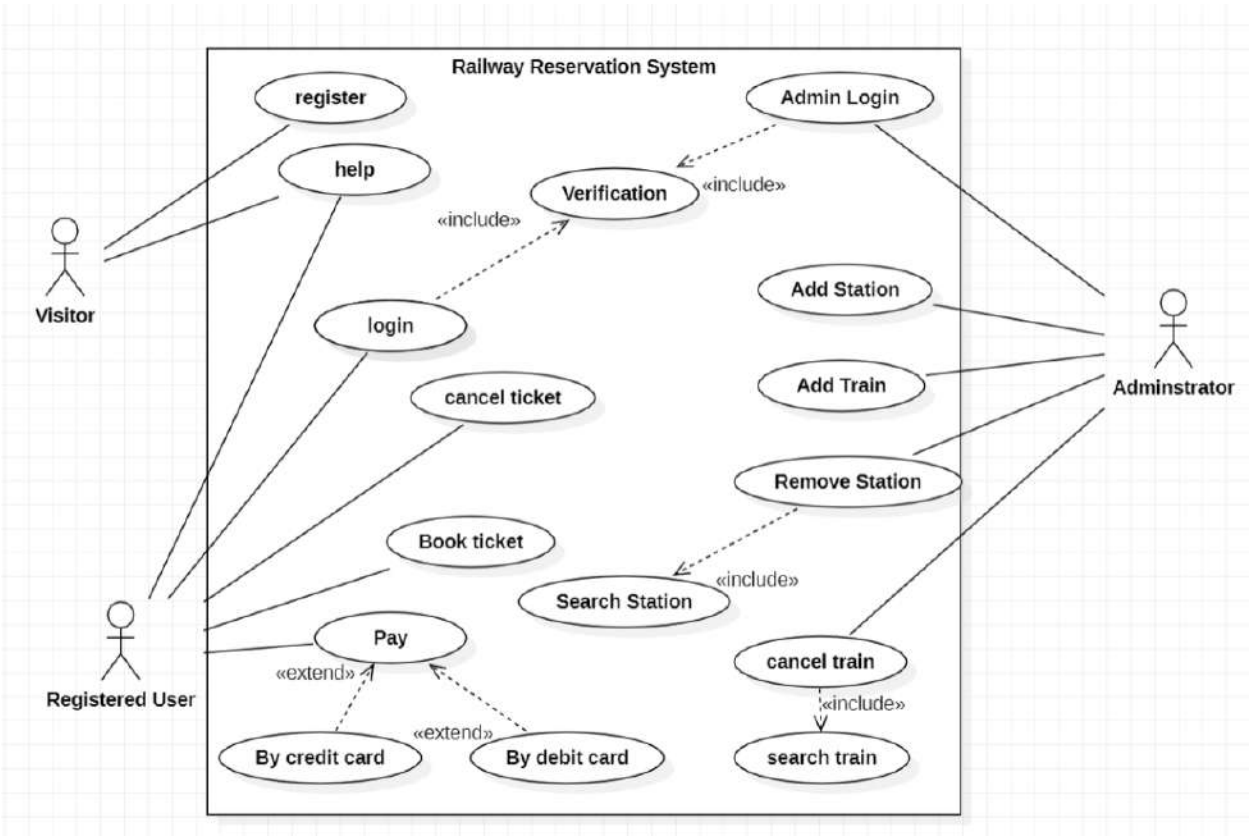
The state diagram contains one nested state and one submachine, which on successful login shows the ChooseTrain details and PaymentSystem procedure. It contains initial state and termination state with ChooseTrain as a nested state including the required simple states. It also has a submachine state named PaymentSystem with initial, termination state along with simple states; Method, Card, Validation, Processing.

6.4 Advanced use case diagram

The advanced use case diagram has extra functionalities which includes extends, includes and generalization. Here both admin login and user login includes verification. Pay by debit card and pay by credit card extends the pay usecase.

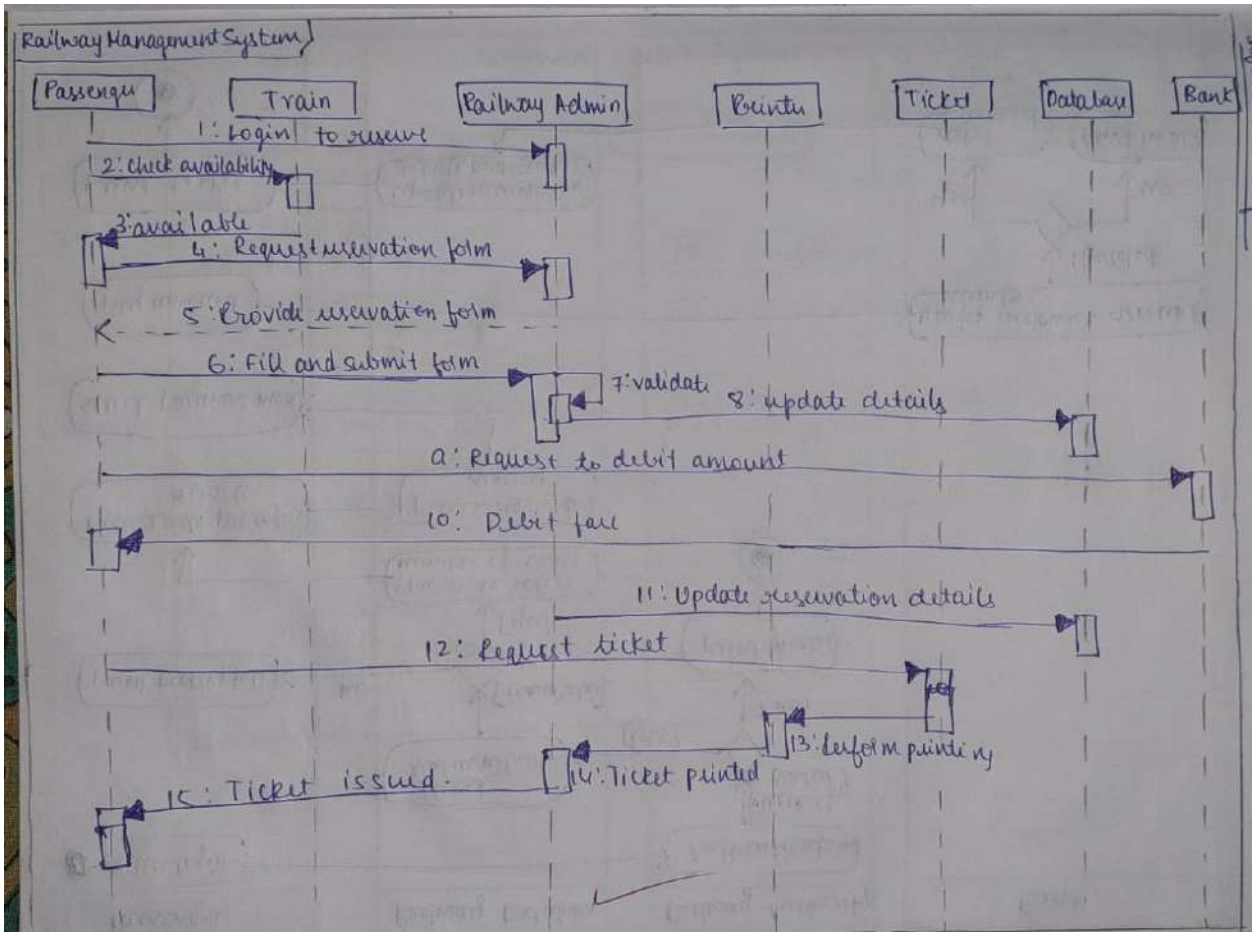
Use-case diagram:

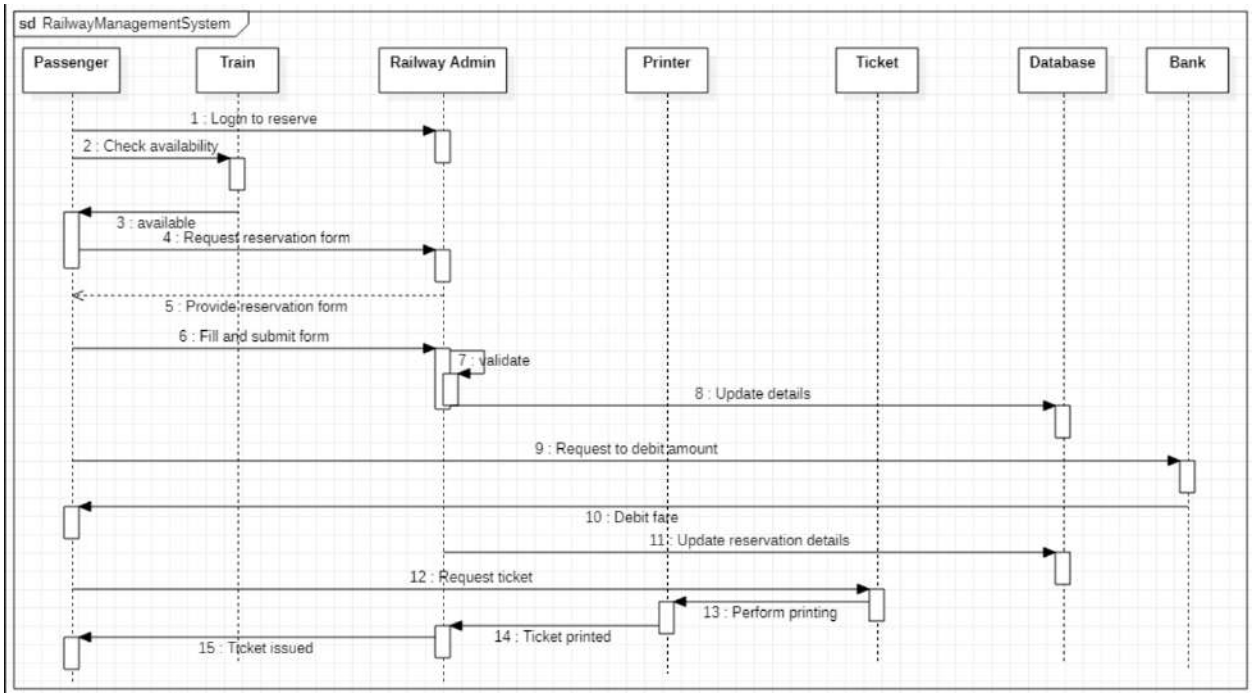




6.5 Advanced Sequence diagram

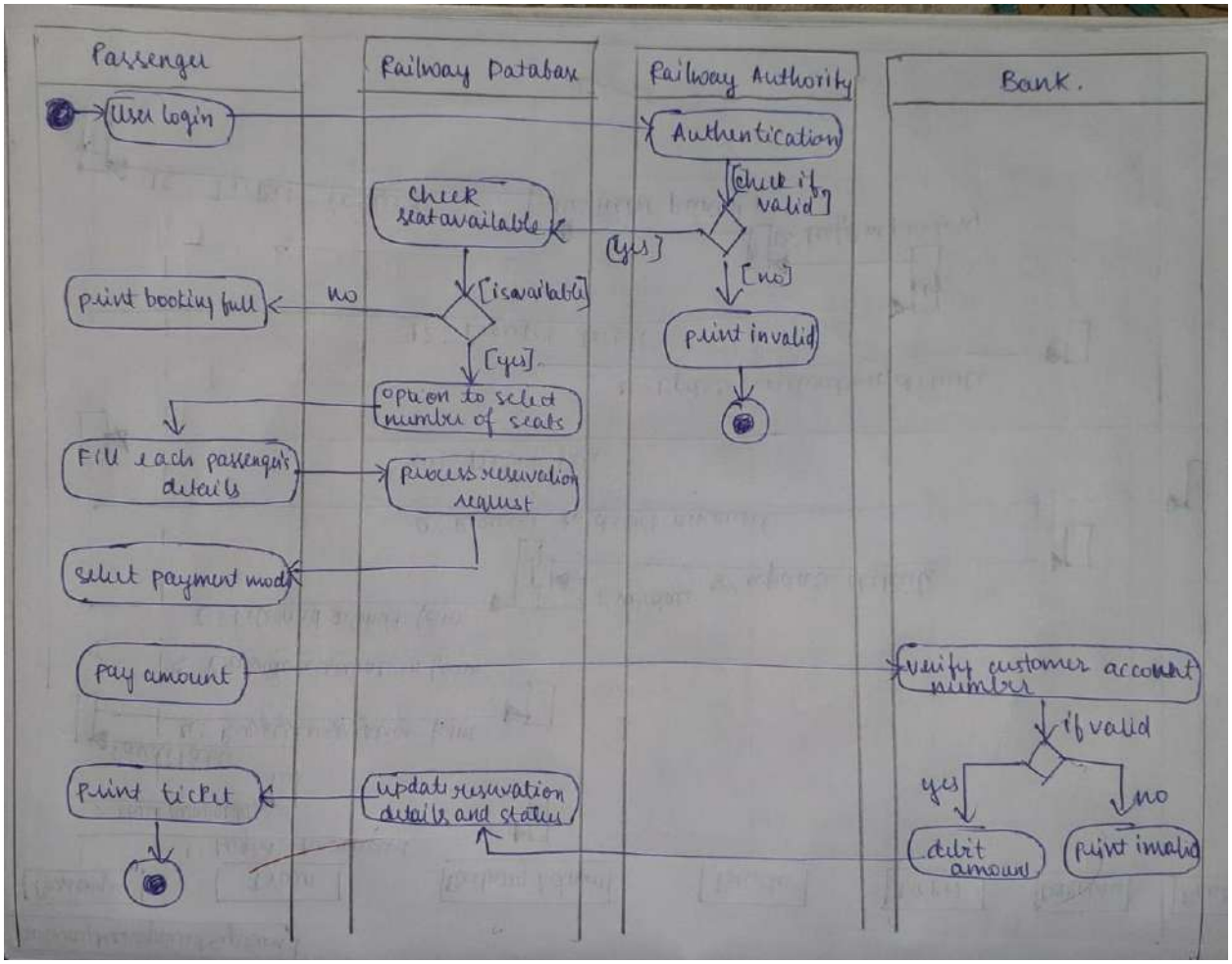
The lifeline is the dotted line and the rectangles represent the period of time the object is executing and is hence called activation. The recursive function of validation is shown by double activation rectangle of validation with self-transition.Reply message is used to to return back to lifelines with the required message. The scenario shown here is customer trying to reserve seats on a train.

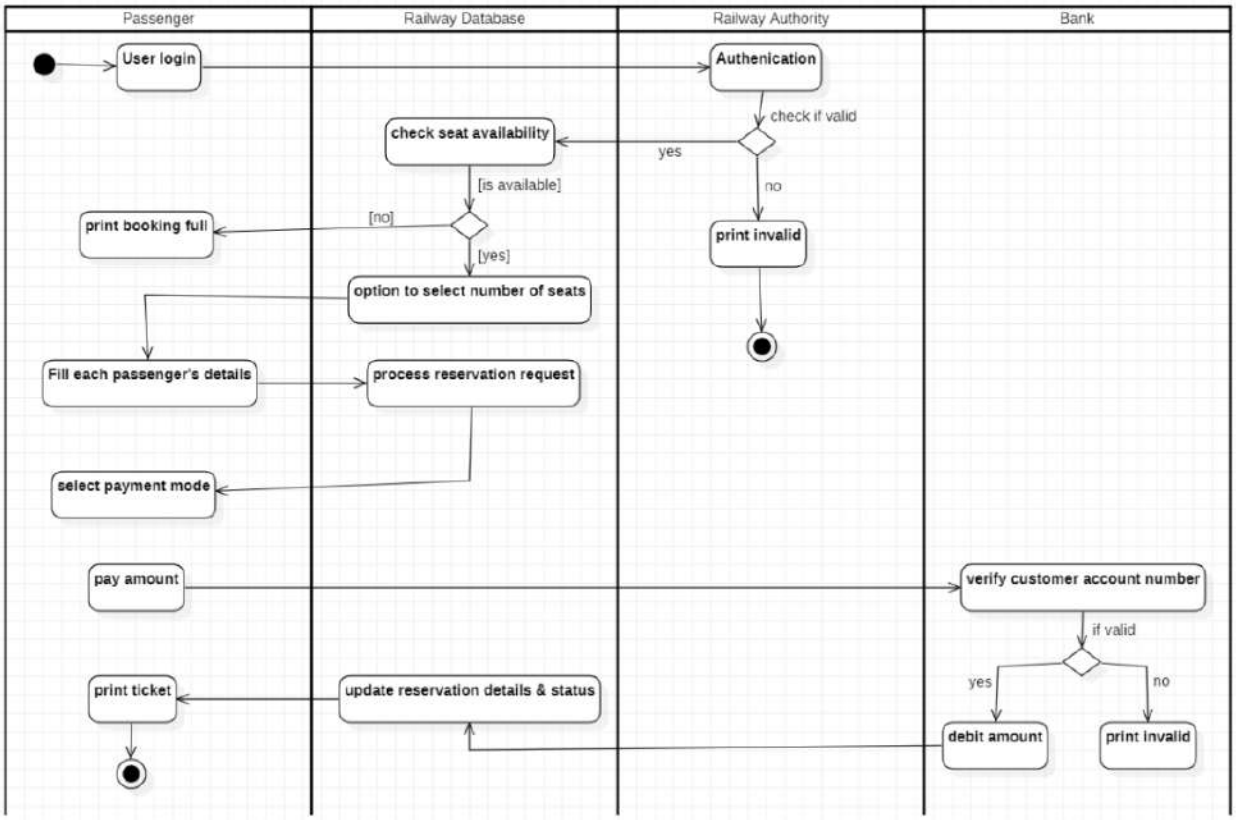

Railway Management System
Passenger
Train
Railway Admin
Printer
Ticket
Database
Bank
1: Login to reserve
2: Check availability
3: available
4: Request reservation form
5: Provide reservation form
6: Fill and submit form
7: validate
8: Update details
9: Request to debit amount
10: Debit fare
11: Update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


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

6.6 Advanced Activity diagram

The advanced activity diagram starts from initiation and in the passenger swimlane, the passenger login activity where a signal is sent to the network for request validation and upon confirmation the control flows to check seat availability activity. There are four swimlanes namely passenger, railway database, railway authority and bank where each one indicates the passenger operations, check seat availability, check validation, confirm payment respectively. Then the control flows to the home page and then termination activities.



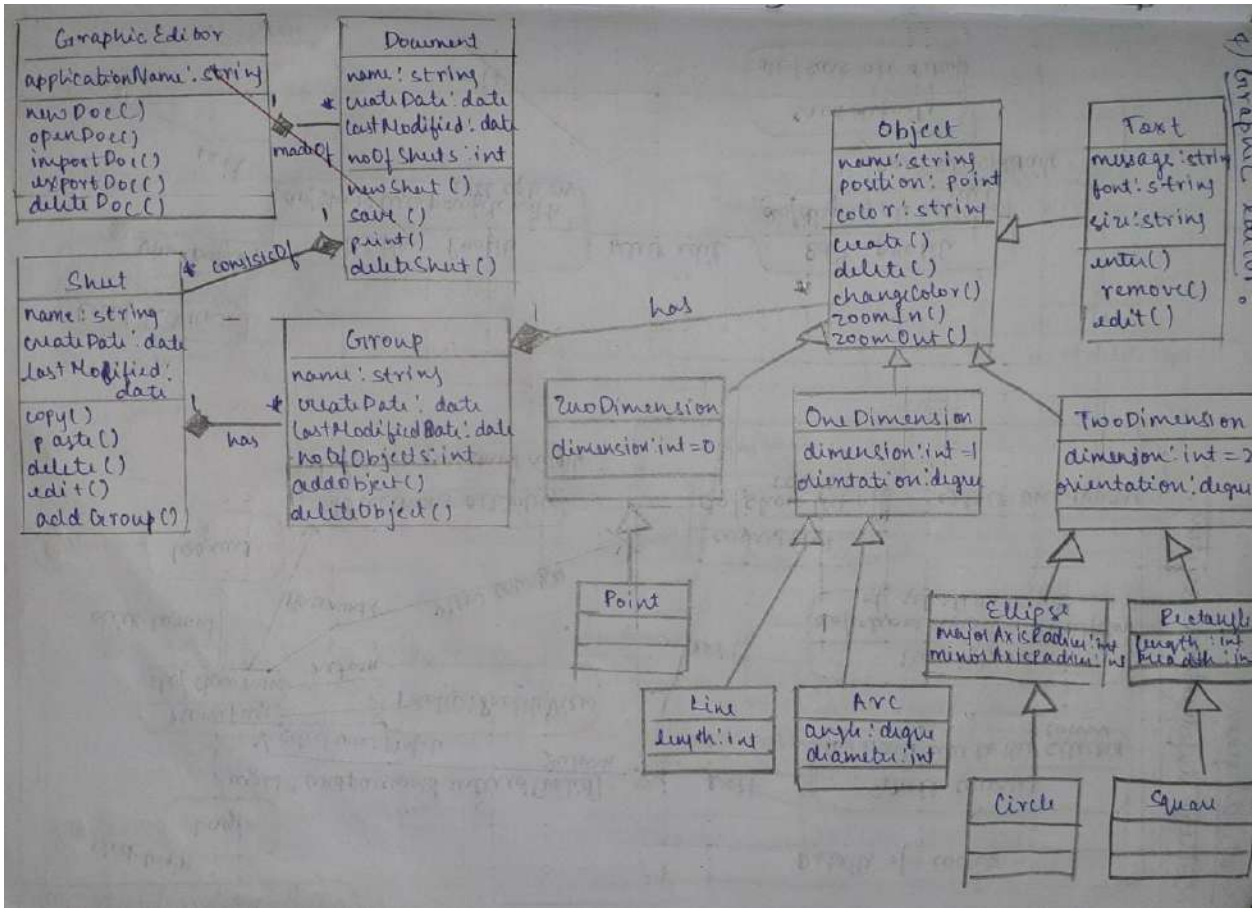

Passenger
Railway Database
Railway Authority
Bank
User login
Authenication
check if valid
check seat availability
yes
no
[is available]
print booking full
[no]
print invalid
[yes]
option to select number of seats
Fill each passenger's details
process reservation request
select payment mode
pay amount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

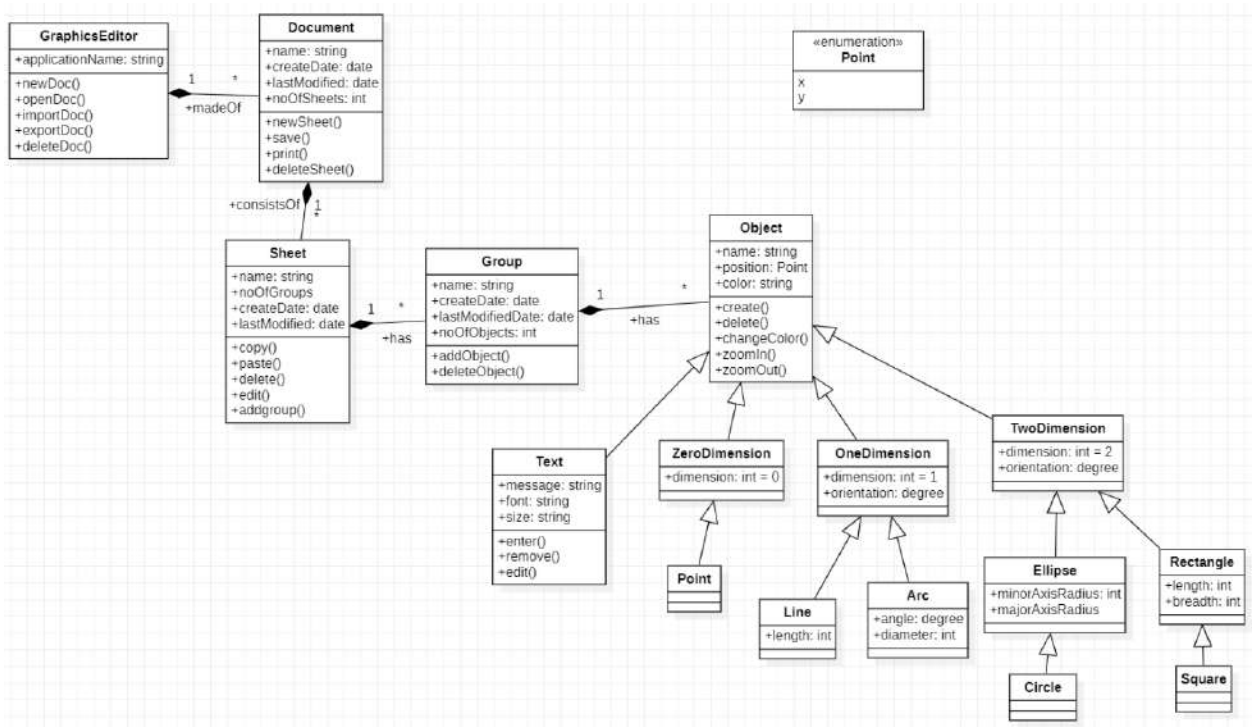
LAB 7:Graphics Editor

7.1 Software Requirement Specification

- The graphical editor consists of a graphical document editor which can be used to createnewdocument,delete documents,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 numbers of drawing objects,which can be created, groupedorformatted.
- The programmer must provide implementations of functions that draw objects andher connections, as well as functions that add and remove connections. The latter function will be handled by a specific event listener. Any changes made in real-time to the underlying model will also be updated in the diagram through a separate 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.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

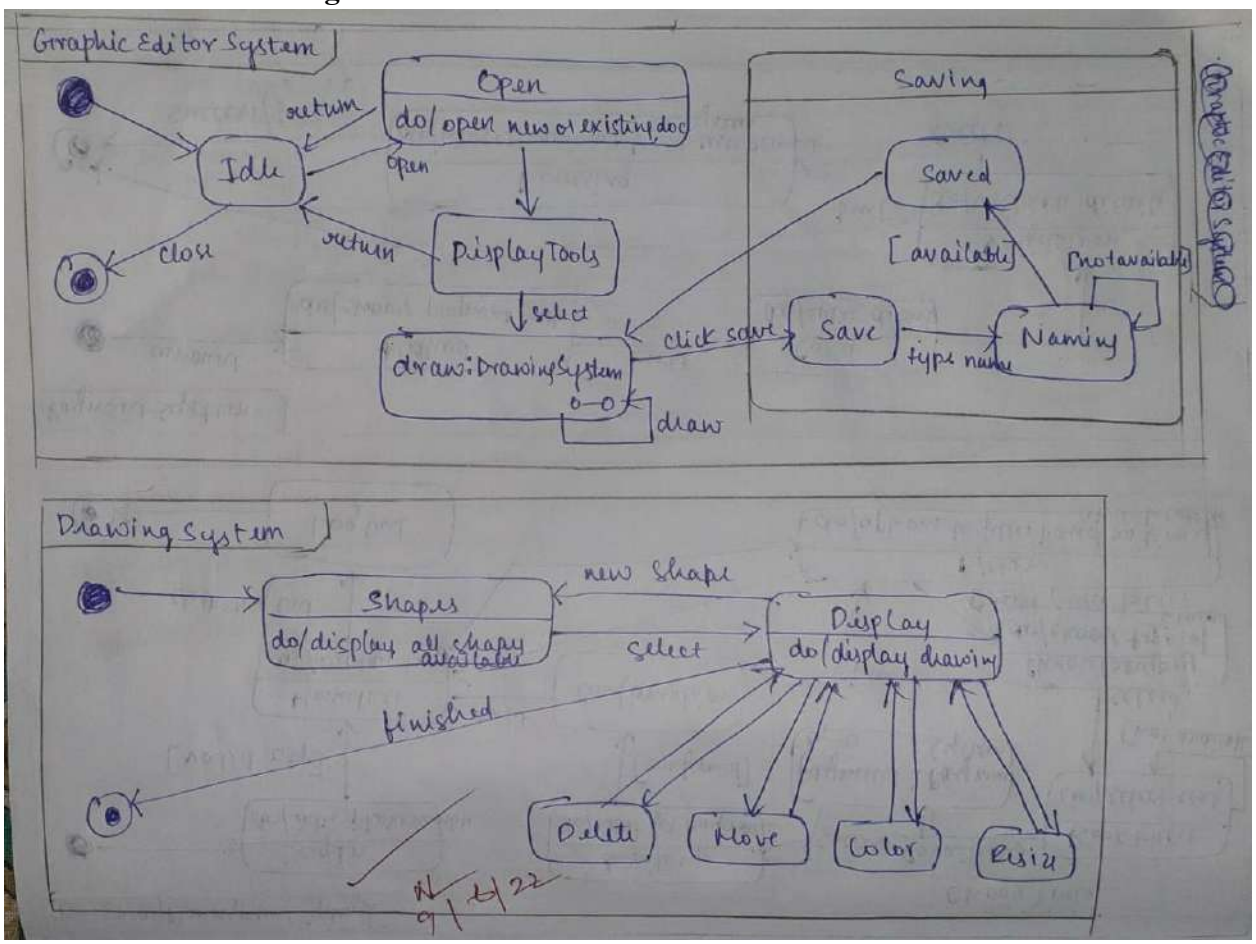
7.2 Advanced Class Diagram

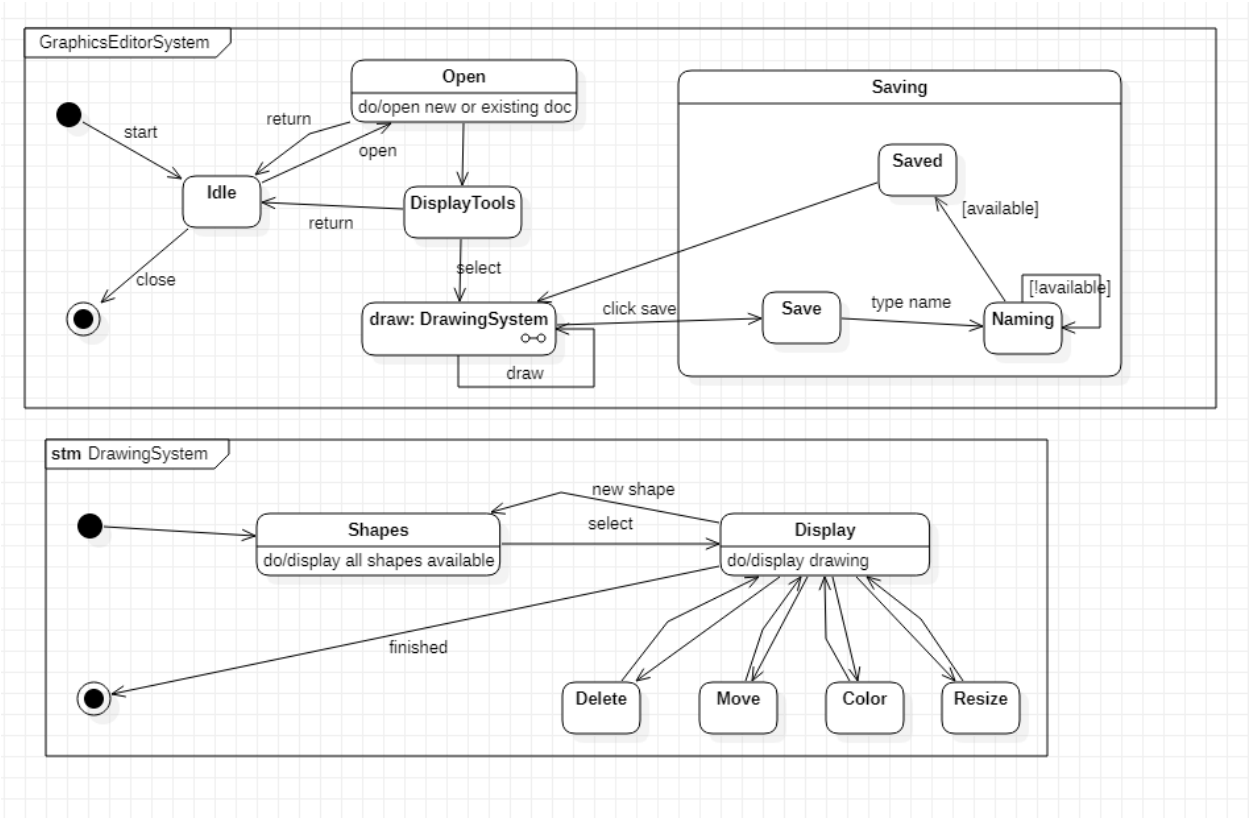

Graphic Editor
applicationName: string
newDoc()
openDoc()
importDoc()
exportDoc()
deleteDoc()
madeOf
Document
name: string
createDate: date
lastModified: date
noOfSheets: int
newSheet()
save()
print()
deleteSheet()
consistsOf
Sheet
name: string
createDate: date
lastModified: date
copy()
paste()
delete()
edit()
addGroup()
has
Group
name: string
createDate: date
lastModifiedDate: date
noOfObjects: int
addObject()
deleteObject()
has
Object
name: string
position: Point
color: string
create()
delete()
changeColor()
zoomIn()
zoomOut()
Text
message: string
font: string
size: string
enter()
remove()
edit()
TwoDimension
dimension: int = 0
OneDimension
dimension: int = 1
orientation: degree
TwoDimension
dimension: int = 2
orientation: degree
Point
Line
length: int
Arc
angle: degree
diameter: int
Ellipse
majorAxisRadius: int
minorAxisRadius: int
Rectangle
length: int
breadth: int
Circle
Square


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
+consistsOf
1
*
Sheet
+name: string
+noOfGroups
+createDate: date
+lastModified: date
+copy()
+paste()
+delete()
+edit()
+addgroup()
1
*
+has
Group
+name: string
+createDate: 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()
ZeroDimension
+dimension: int = 0
Point
OneDimension
+dimension: int = 1
+orientation: degree
Line
+length: int
Arc
+angle: degree
+diameter: int
TwoDimension
+dimension: int = 2
+orientation: degree
Ellipse
+minorAxisRadius: int
+majorAxisRadius
Rectangle
+length: int
+breadth: int
Circle
Square

- Graphics Editor class is made up of the composition of Document class as document class is there particularly for this class.
- And the Document class is made up of the composition of the Sheet class and sheet of group class and group of object class as one class cannot exist without another class.
- Object class is generalized into text, zeroDimension, One dimension and two dimension class.
- Zero dimension is generalized into point.one dimension class into line and arc.two dimension classinto ellipse and rectangle class and further ellipse and rectangle into circle and square respectively.

7.3 Advanced State Diagram

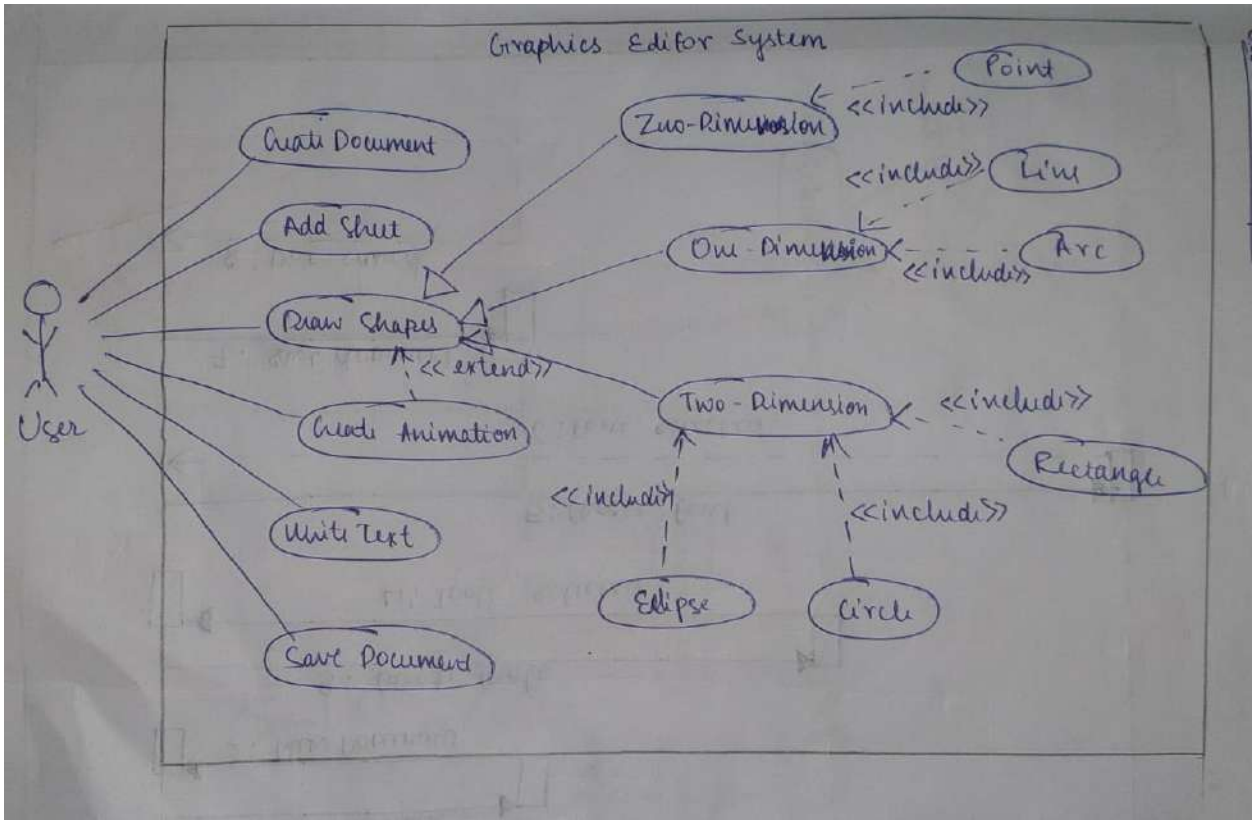


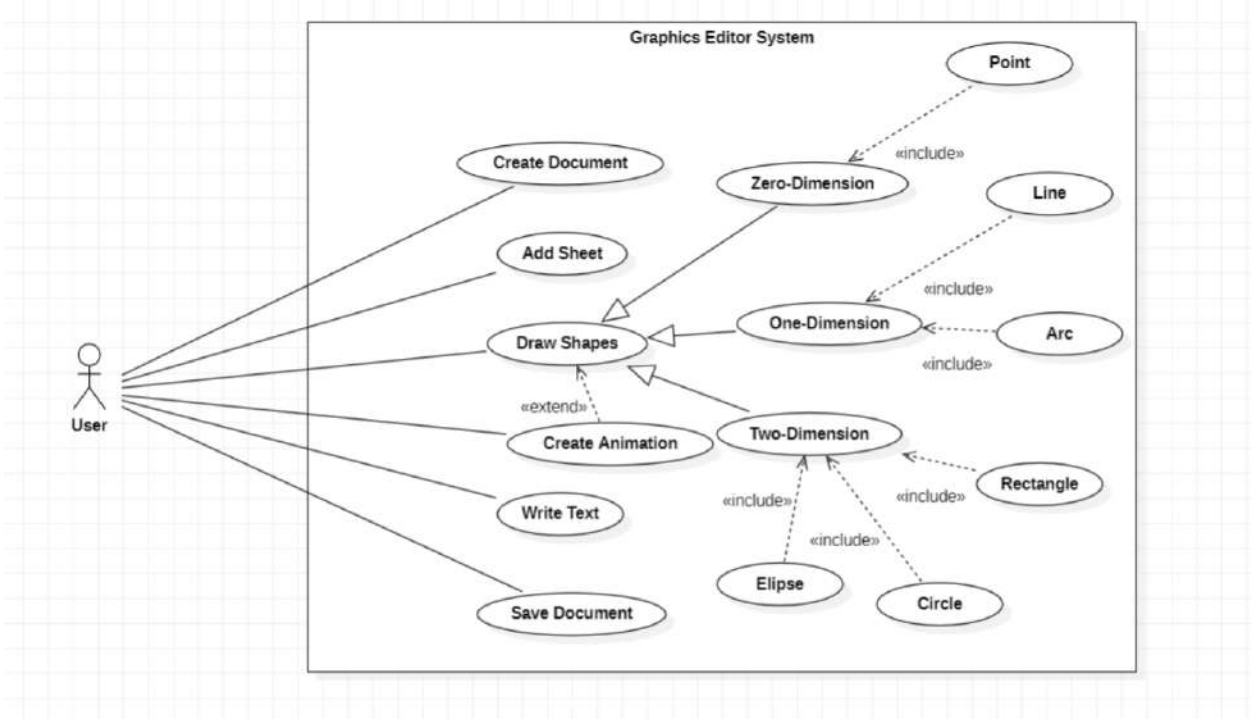


The state diagram contains one nested state and one submachine, which on successful login shows the Saving procedure and DrawingSystem procedure. It contains initial state and termination state with Saving as a nested state including the required simple states. It also has a submachine state named DrawingSystem with initial, termination state along with simple states; Shapes, Display and format each shape.

7.4 Advanced use case diagram

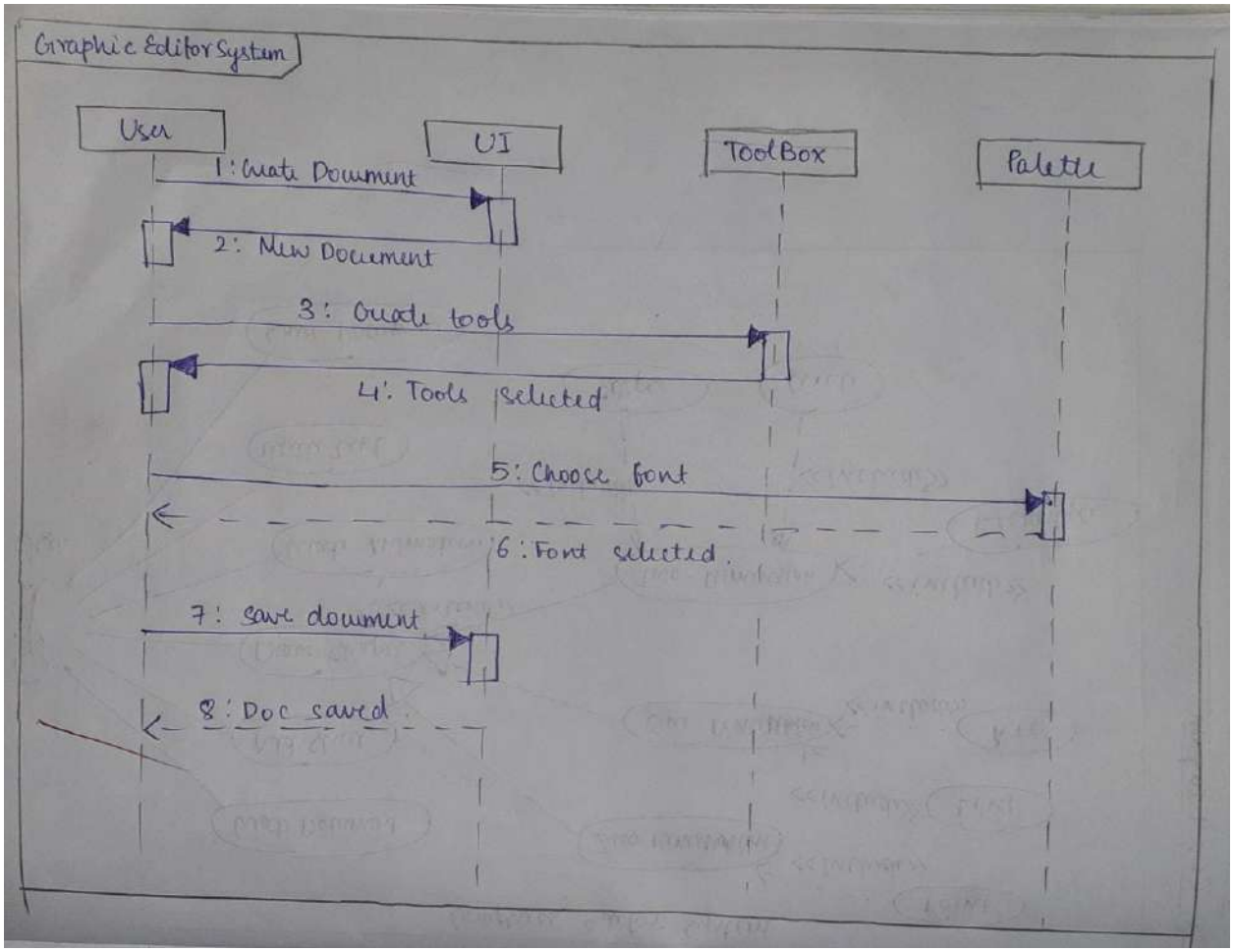
The advanced use case diagram has extra functionalities which includes extends, includes and generalization. The edit document use case extends new document use case, delete document use case extends new document use case, graphic tools use case extends new document use case, new document use case includes save document use case. Insert, delete and color is generalized to super class graphics tools.

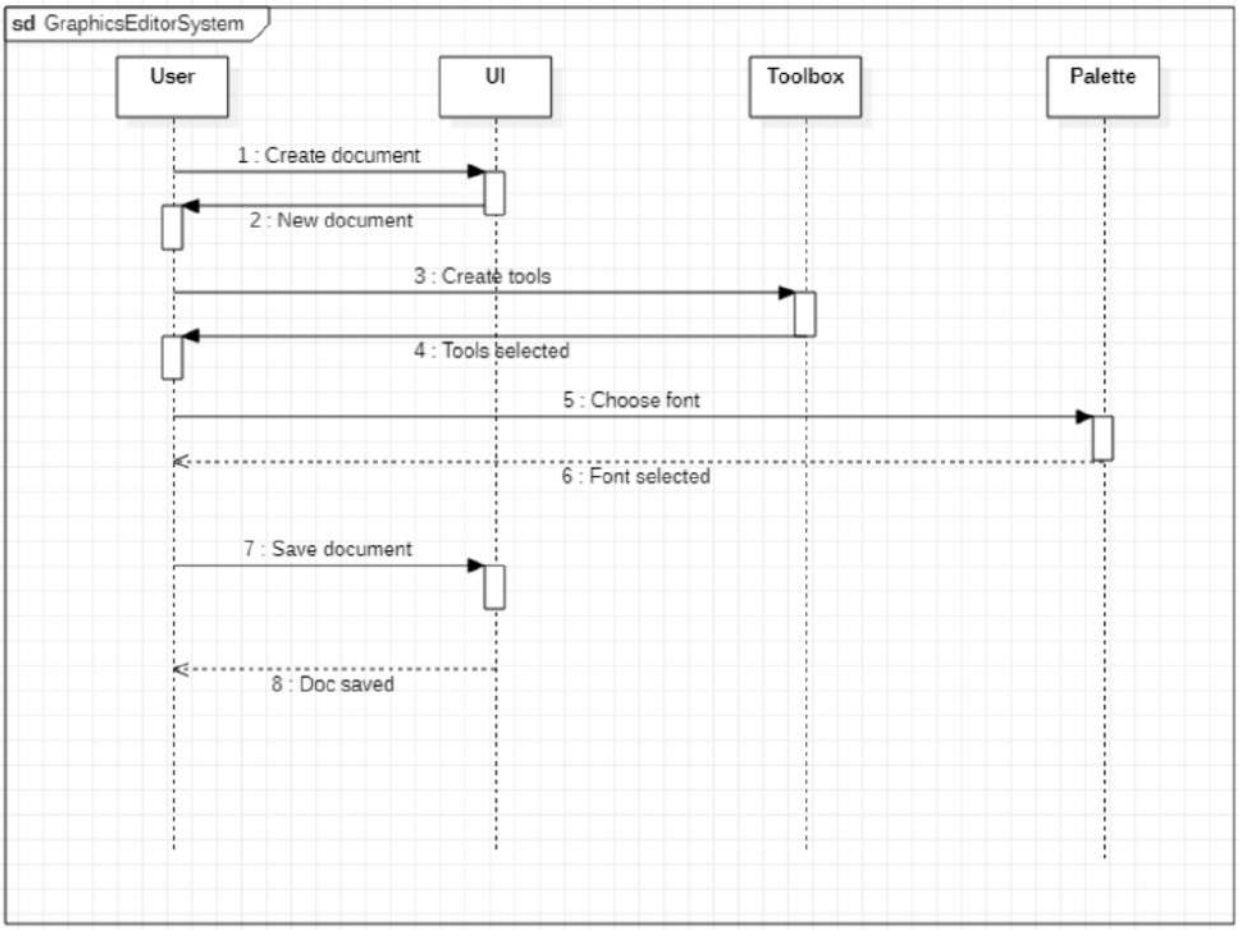




7.5 Advanced Sequence diagram

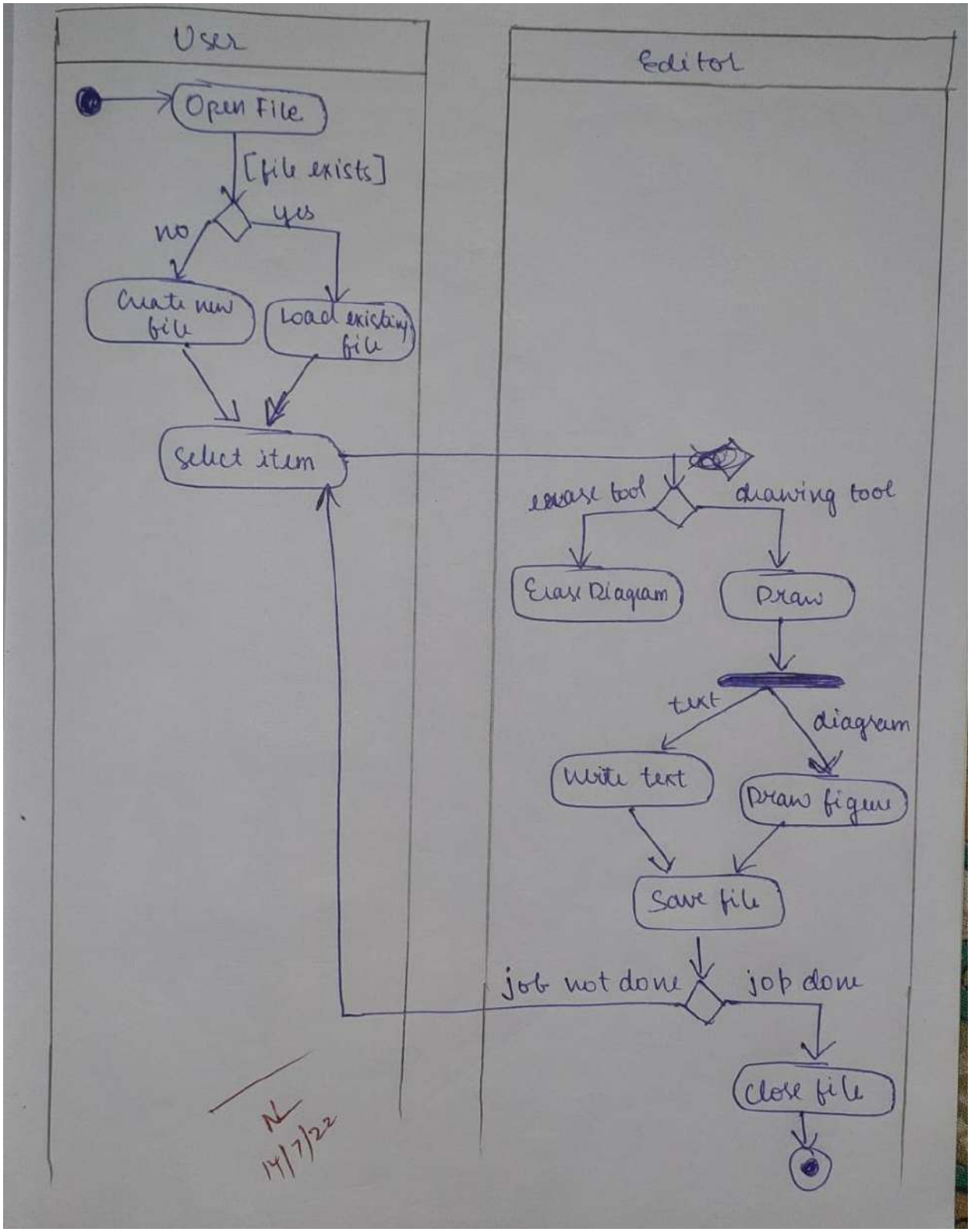
The lifeline is the dotted line and the rectangles represent the period of time the object is executing and is hence called activation. Reply message is used to to return back to lifelines with the required message. The scenario shown here is user creating a document and saving it.

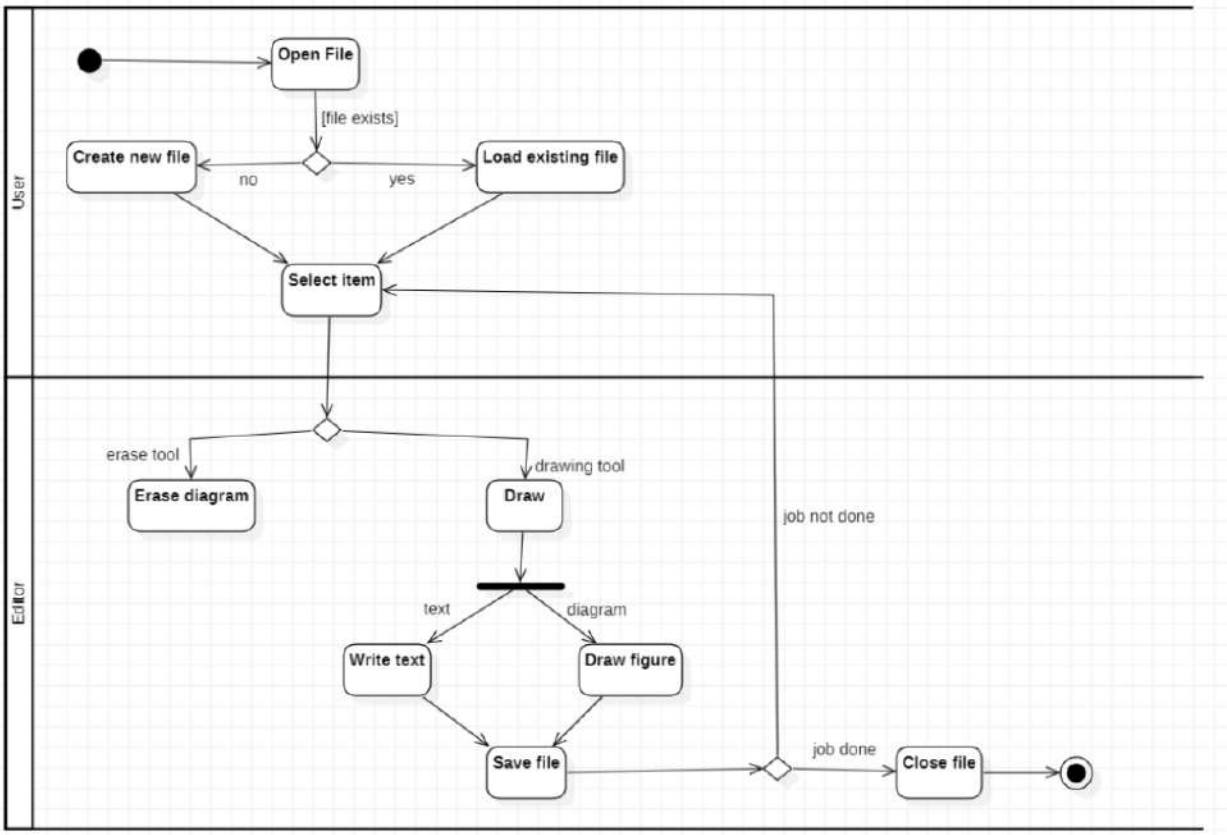




7.6 Advanced Activity diagram

The advanced activity diagram starts from initiation and in the user swimlane, the user login activity where a signal is sent to the network for request validation and upon confirmation the control flows to open file activity. There are two horizontal swimlanes namely user and editor where each one indicates the user operations and drawing a diagram respectively. Then the control flows to the close file activity and then termination activities.


User
Editor
Open File
[file exists]
no
yes
Create new file
Load existing file
Select item
erase tool
drawing tool
Erase Diagram
Draw
text
diagram
Write text
Draw figure
Save file
job not done
job done
Close file
14/7/22


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