IT2020 – Software Engineering

Year 2 - Semester I (June) - 2021

Online vehicle service and fuel station management system

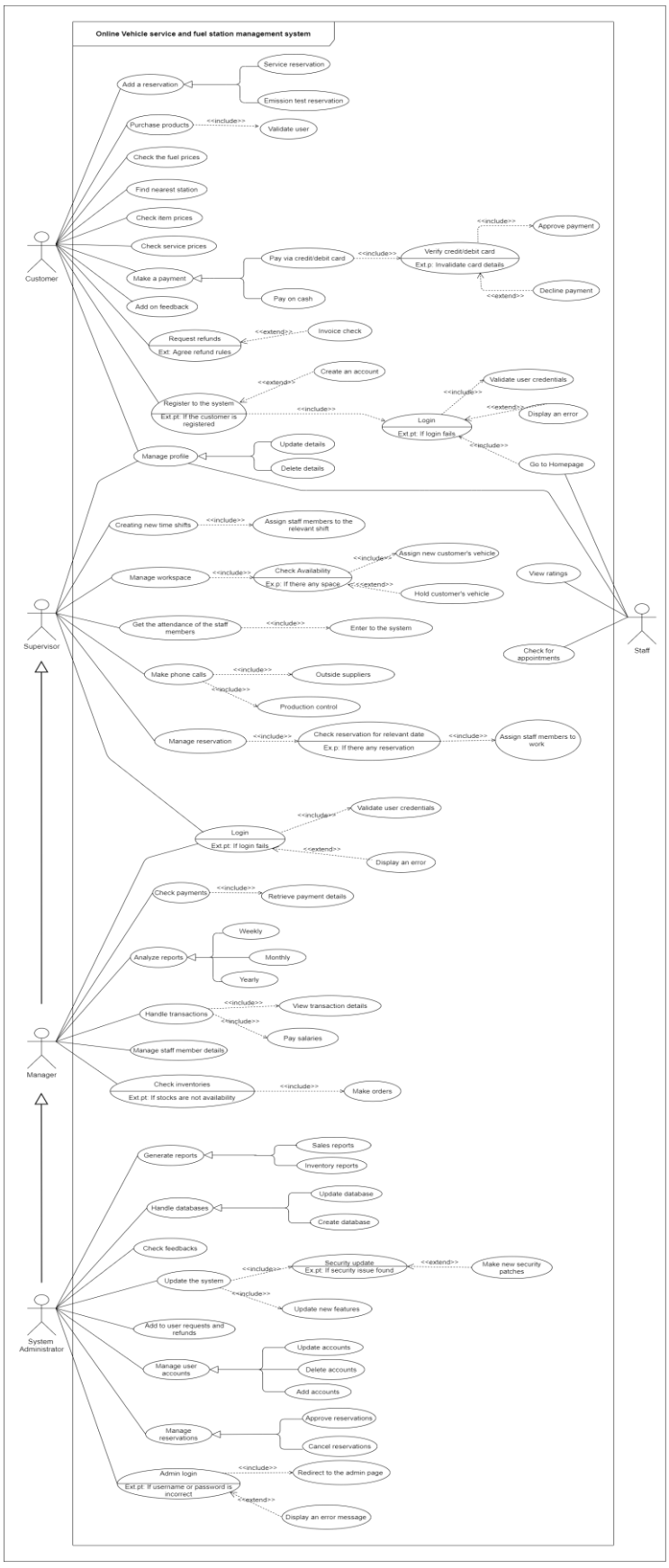
Group Details

Batch: 2

SE/OOP Group Number: 2021_June_Malabe_B2_11

	Student Registration Number	Student Name
1	IT20625016	Nanayakkara H. M. S. Y.
2	IT20606060	Wanni Arachchige H. S.
3	IT20297640	Piyasinghe H. K. N. Y.
4	IT20150648	Jayakody J. A. M. G.

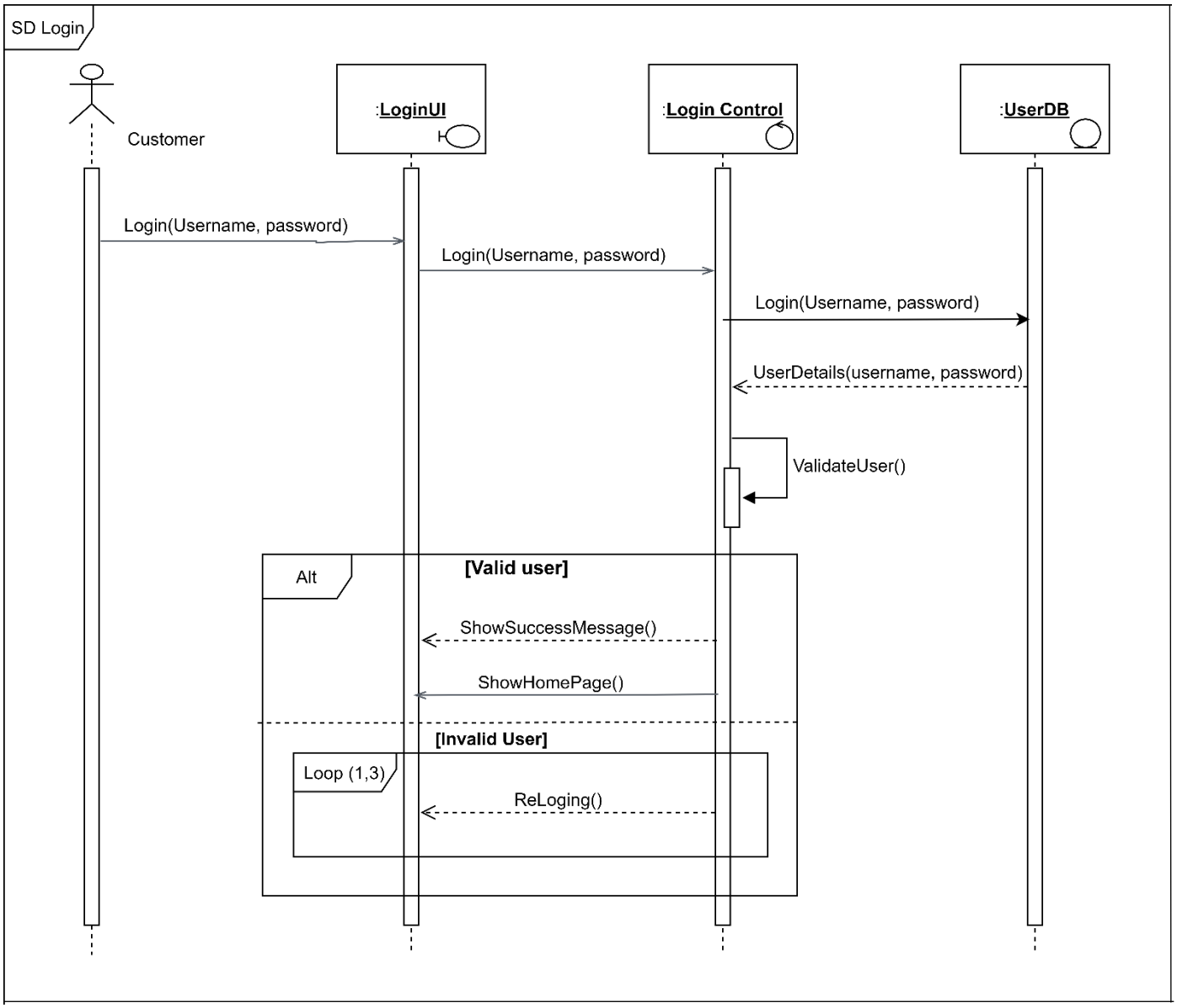
Use Case Diagram

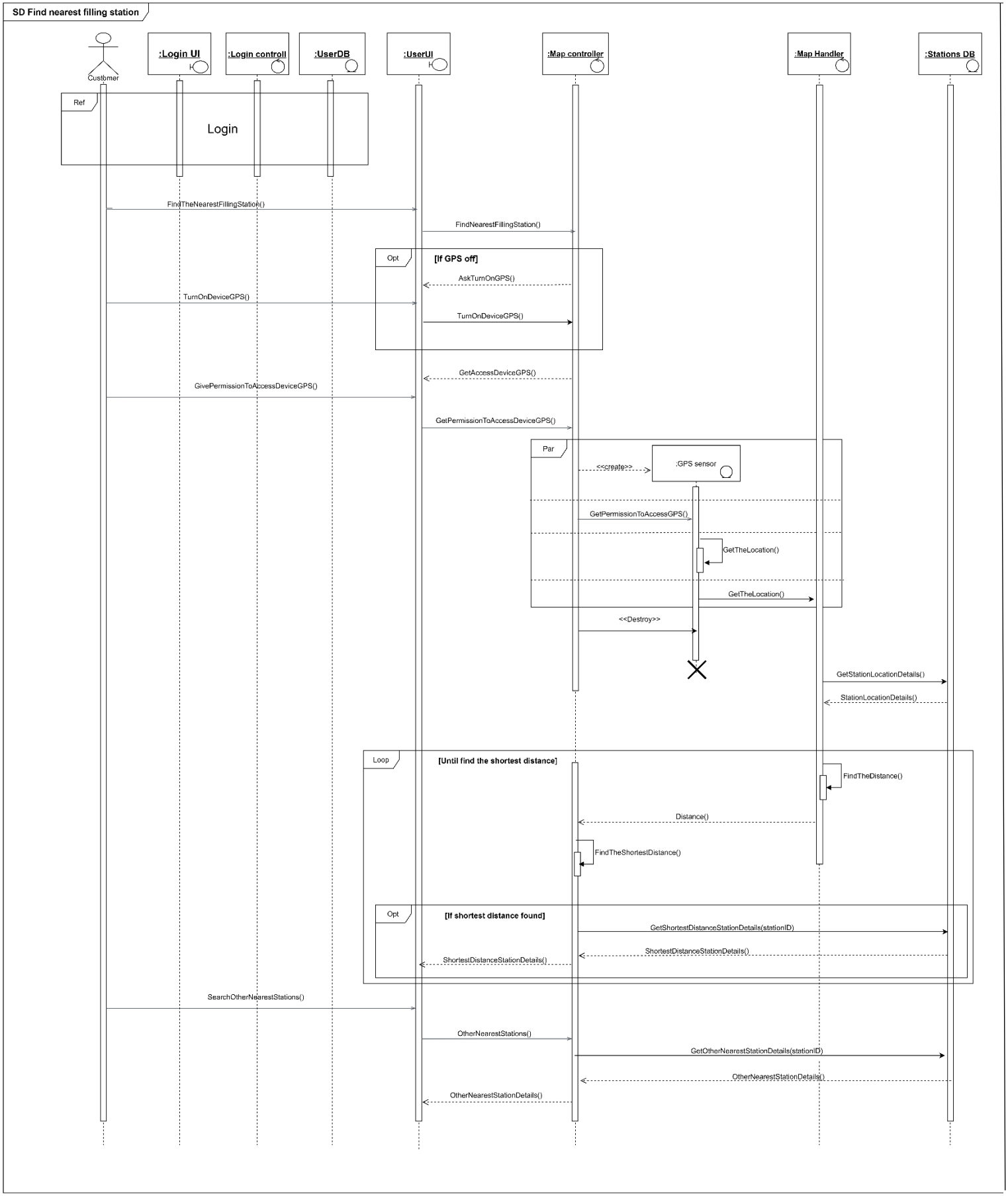


1. Use Case Scenario

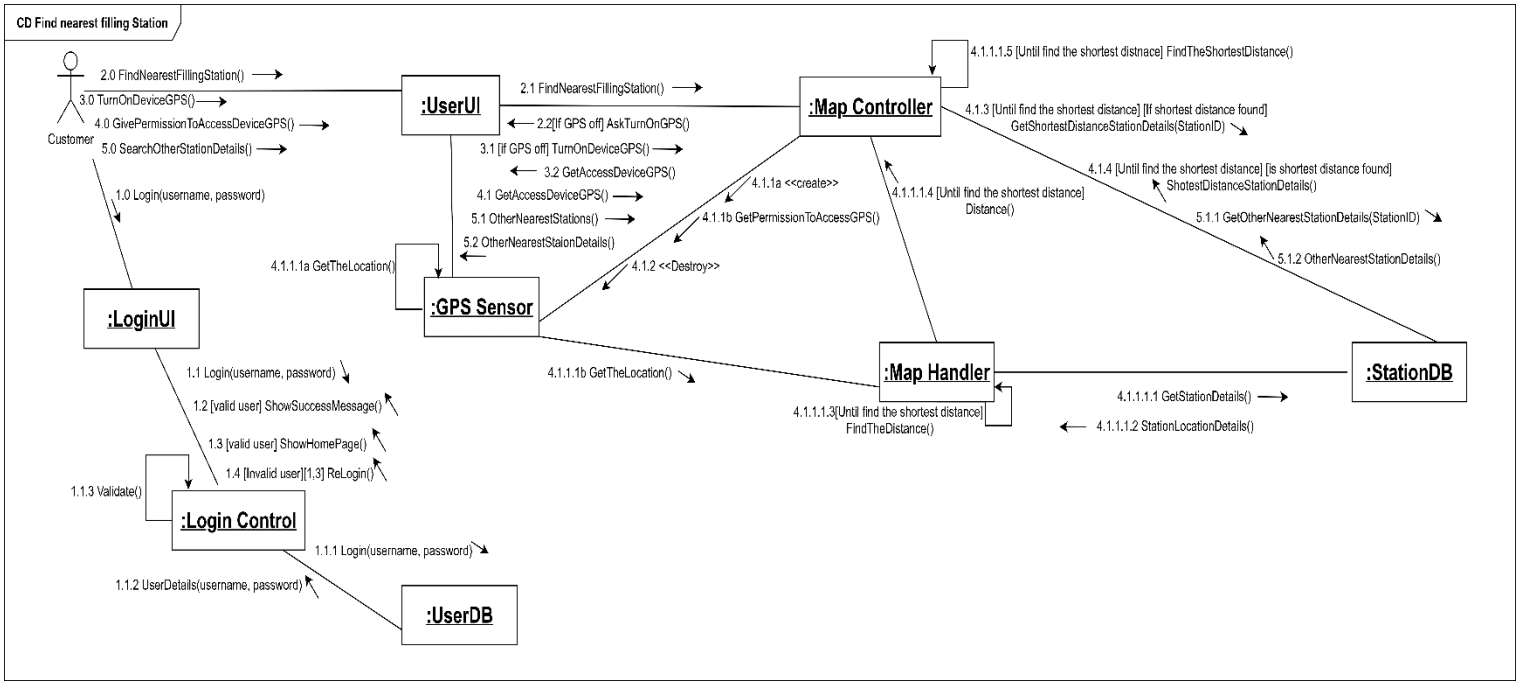
Number	1
Name	Find the nearest filling station according to the current location
Summery	Get the customer's current location and find the filling station which has the shortest distance
Precondition	Customer should turn on location and give permission to the system to get customer's current location
Postcondition	Display the nearest filling station with distance and suggest some other filling station
Primary actor	Customer
Priority	4
Main Scenario	1. Go to the login page. 2. Customer login to the system using username and password. 3. Select find nearest filling station function. 4. Get the customer's current location. 5. Identify the location, routes, near to the customer's location. 6. Search the database and find the filling station that situated in customer's location. 7. Identify the filling stations and calculate the distance to each station from the customer's location. 8. Find the shortest distance. 9. Display if first and display other distance according to the ascending order. 10. Customer can select one of the stations from the station list. 11. Customer can turn on the directions and follow the showing route.
Extension	4a. Create a GPS sensor object. 4b. If customer turn on the location (GPS). 4c. Ask for permission to get access the device GPS. 4d. If customer turn off the location (GPS). 4e. Show a notification to turn on the device location. 4f. After that ask for permission to get access device GPS. 4g. Return the current location. 4h. Destroy the GPS sensor object. 8a. Get the distance one by one and compare the distances 8b. get the minimum value

Sequence Diagram




SD Find nearest filling station
Customer
:Login UI
:Login controll
:UserDB
:UserUI
:Map controller
:Map Handler
:Stations DB
Ref
Login
FindTheNearestFillingStation()
FindNearestFillingStation()
Opt
[If GPS off]
AskTurnOnGPS()
TurnOnDeviceGPS()
TurnOnDeviceGPS()
GetAccessDeviceGPS()
GivePermissionToAccessDeviceGPS()
GetPermissionToAccessDeviceGPS()
Par
<<create>>
:GPS sensor
GetPermissionToAccessGPS()
GetTheLocation()
GetTheLocation()
<<Destroy>>
GetStationLocationDetails()
StationLocationDetails()
Loop
[Until find the shortest distance]
FindTheDistance()
Distance()
FindTheShortestDistance()
Opt
[If shortest distance found]
GetShortestDistanceStationDetails(stationID)
ShortestDistanceStationDetails()
ShortestDistanceStationDetails()
SearchOtherNearestStations()
OtherNearestStations()
GetOtherNearestStationDetails(stationID)
OtherNearestStationDetails()
OtherNearestStationDetails()

Communication Diagram

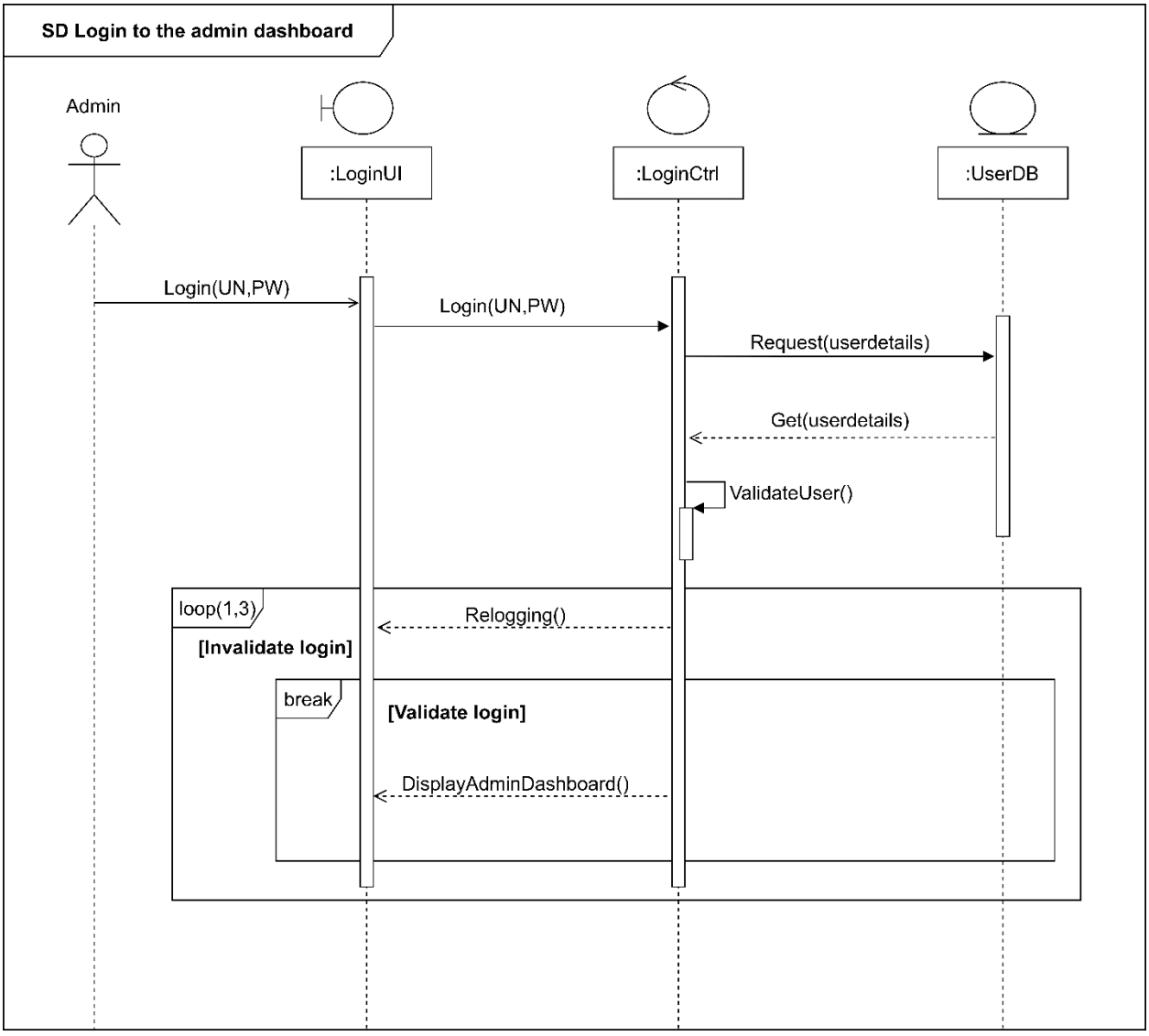


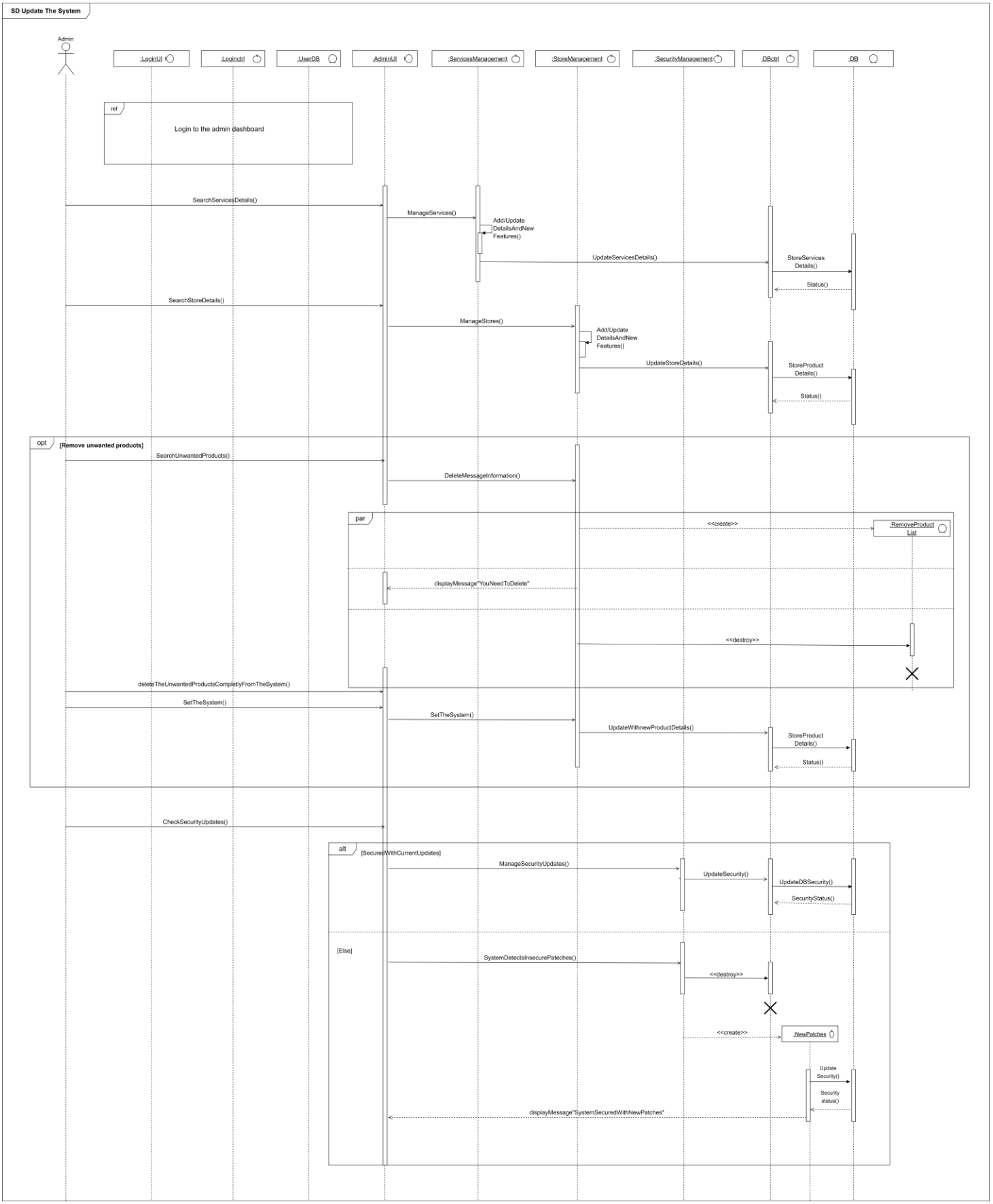
1) Use Case Scenario

Number	2	
Name	Update the system	
Summary	Updating the all new features successfully and ensure that the system is secured.	
Priority	**5**	
Pre-conditions	System administrator has logged in to the system.	
Post-conditions	The system has been updated by new or modified items and services.	
Primary Actor	System Administrator	
Trigger	The admin has chosen to update the database system.	
Main Scenario	**Step**	**Action**
	1	Admin enter login credentials to the system.
	2	System validate login credentials.
	3	Admin searches the details that he wants to update in the System by using relevant field.
	4	Admin starts to update all relevant systems.
	5	Admin goes to the services system.
	6	Admin manage and update services details.
	7	Admin manage store management system by adding new features and products.
	8	Admin searches products names that need to remove from the system.
	9	Admin removes the product's details from the store management system.
	10	Admin set all the system.
	11	Admin access the product's database and delete the unwanted products list.
	12	Admin goes to the security system and update the system.
Extensions	**Step**	**Branching Action**
	1.1	If password or the user name is incorrect , the system displays an error massage

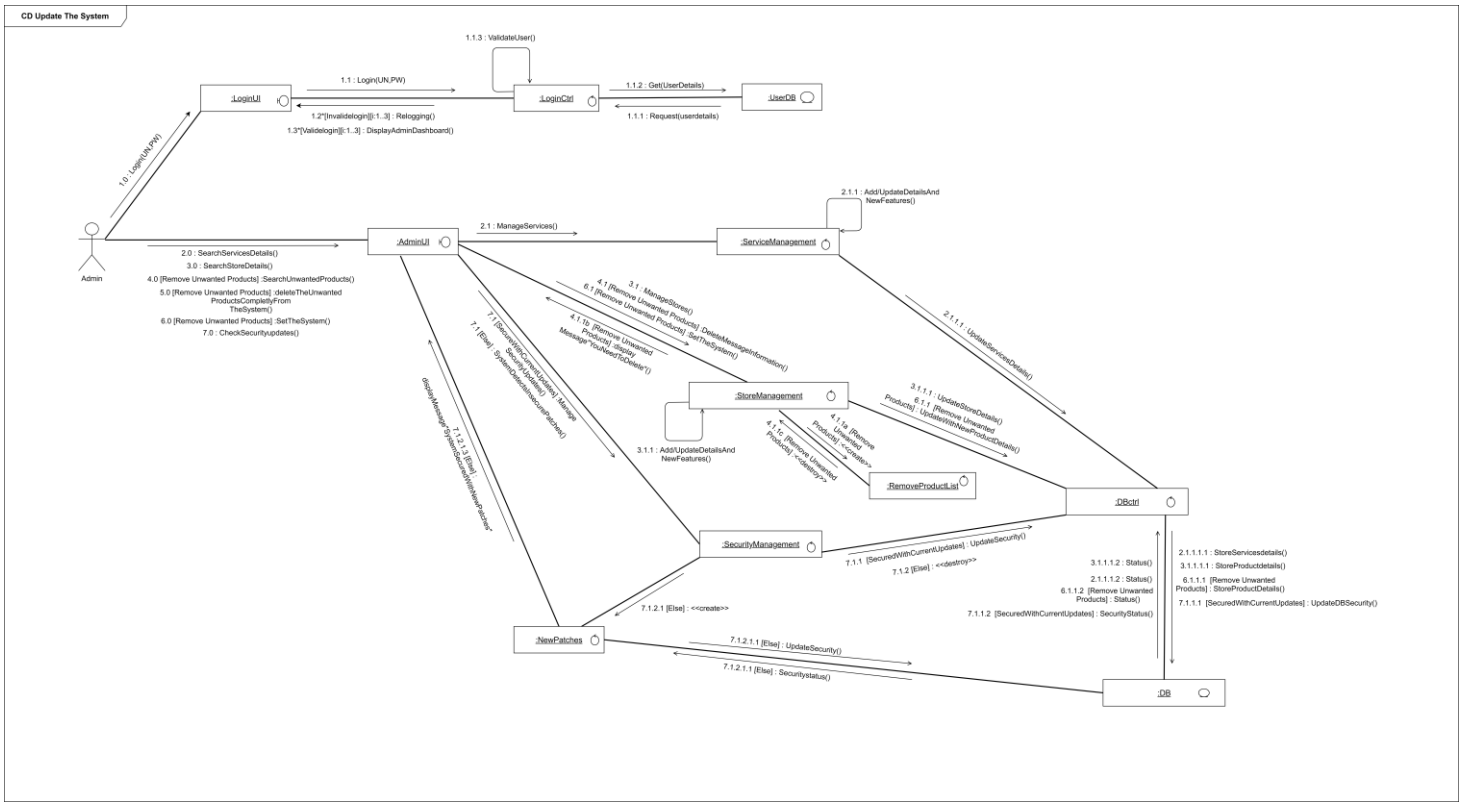
	2.1	System shows error message for invalid reservation.
	8.1	System displays that results.
	9.1	System displays a message “You need to delete”.
	10.1	Create a new product list DB.
	11.1	If security issues found. a. The system detects insecure system. b. The system destroy insecure patches. c. Create new security patches for insecure system.

Sequence Diagram




SD Update The System
Admin
:LoginUI
:Loginctrl
:UserDB
:AdminUI
:ServicesManagement
:StoreManagement
:SecurityManagement
:DBctrl
:DB
ref
Login to the admin dashboard
SearchServicesDetails()
ManageServices()
Add/Update DetailsAndNew Features()
UpdateServicesDetails()
StoreServices Details()
Status()
SearchStoreDetails()
ManageStores()
Add/Update DetailsAndNew Features()
UpdateStoreDetails()
StoreProduct Details()
Status()
opt
[Remove unwanted products]
SearchUnwantedProducts()
DeleteMessageInformation()
par
<<create>>
:RemoveProduct List
displayMessage"YouNeedToDelete"
<<destroy>>
deleteTheUnwantedProductsCompletlyFromTheSystem()
SetTheSystem()
SetTheSystem()
UpdateWithnewProductDetails()
StoreProduct Details()
Status()
CheckSecurityUpdates()
alt
[SecuredWithCurrentUpdates]
ManageSecurityUpdates()
UpdateSecurity()
UpdateDBSecurity()
SecurityStatus()
[Else]
SystemDetectsInsecurePateches()
<<destroy>>
<<create>>
:NewPatches
Update Security()
Security status()
displayMessage"SystemSecuredWithNewPatches"

Communication Diagram

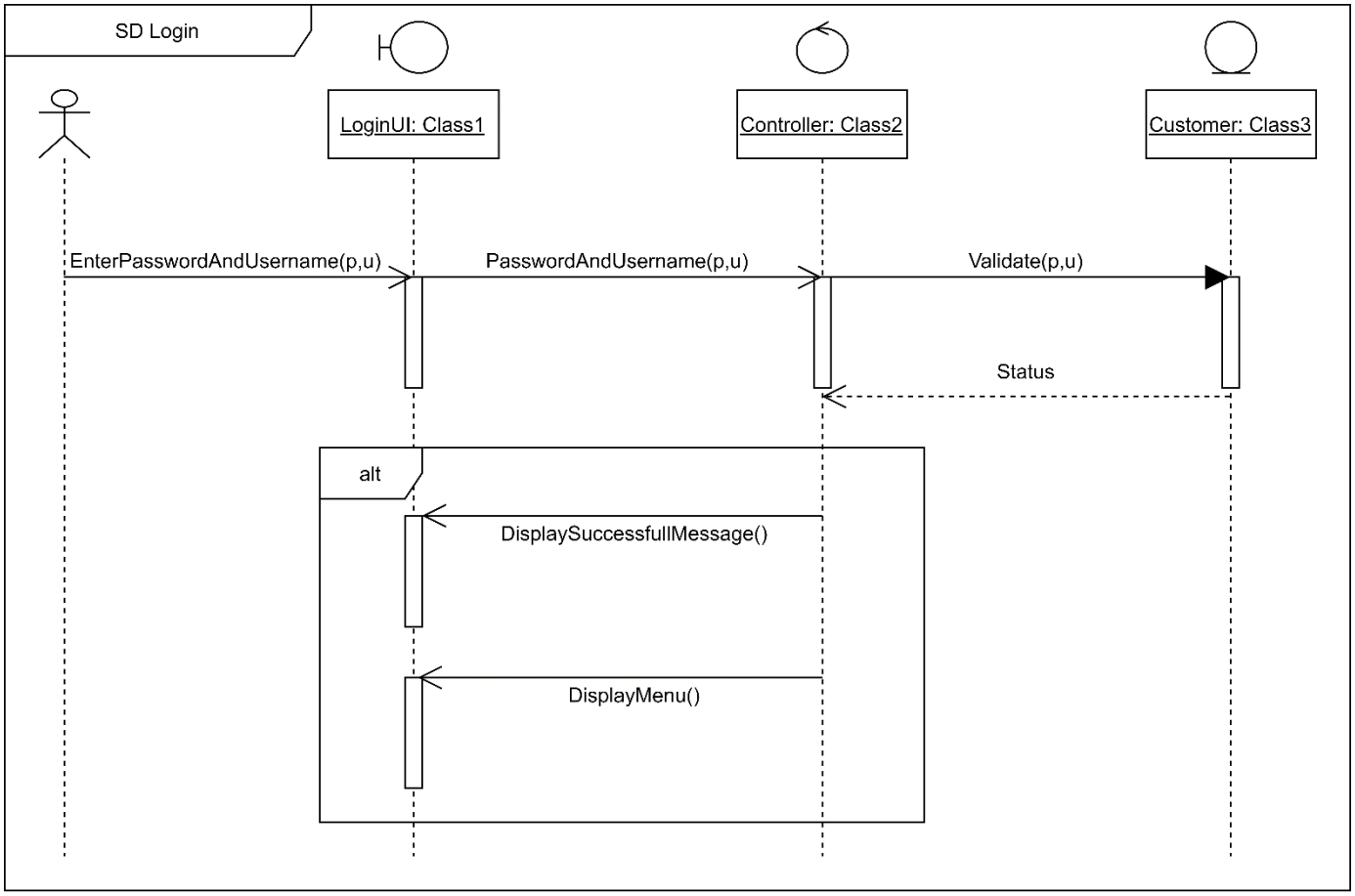


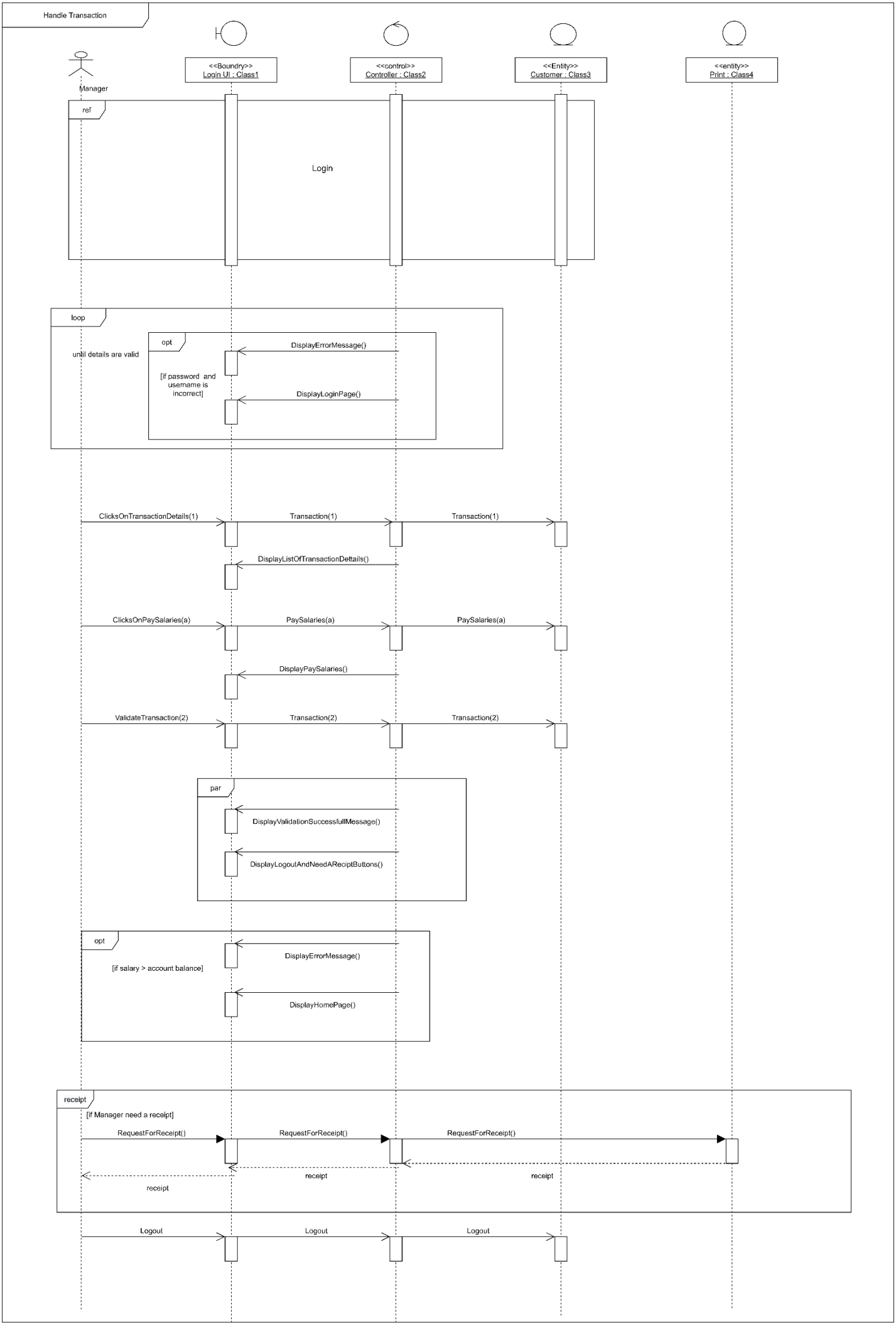
IT20297640 Piyasinghe H. K. N. Y.

1)Use case scenario

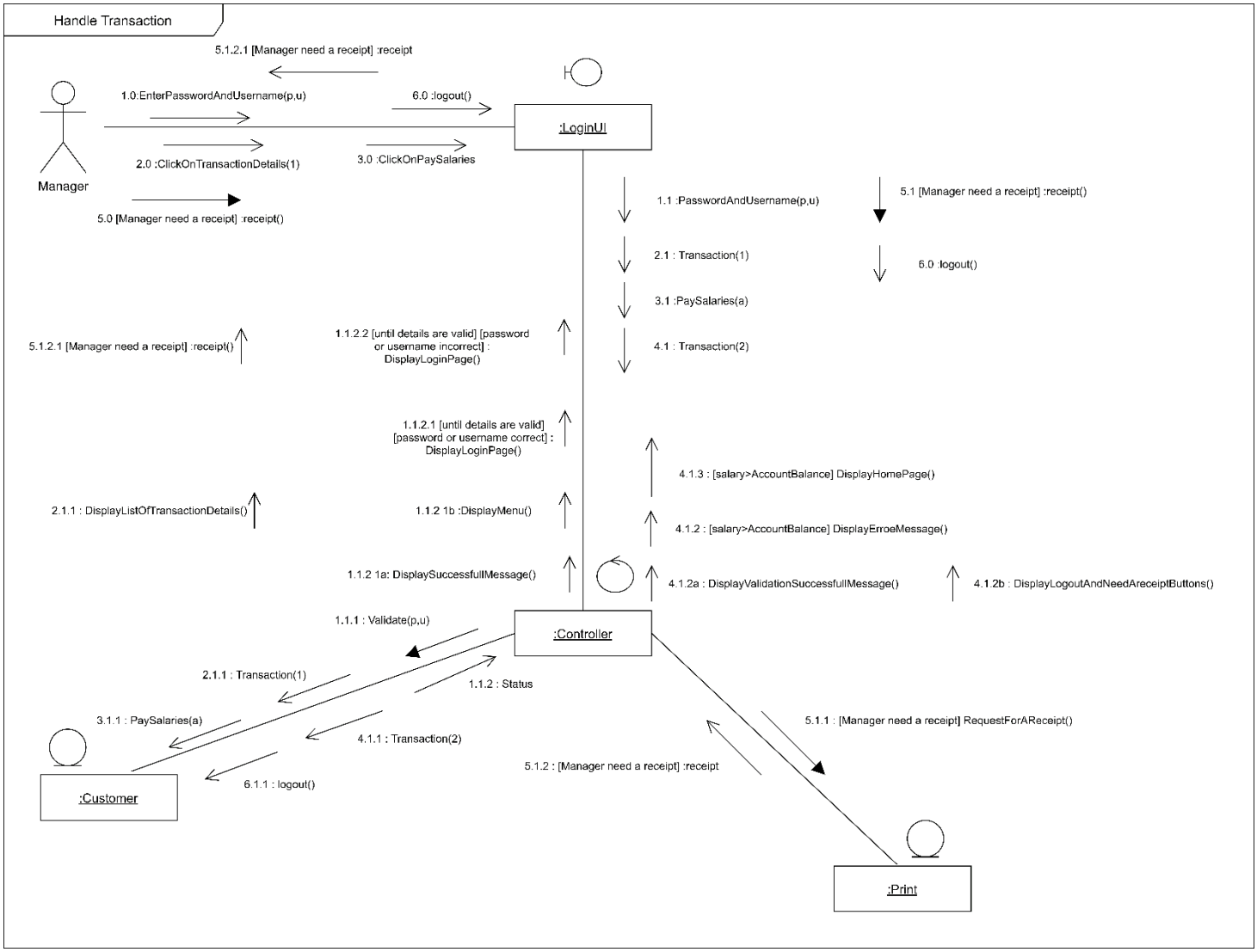
Number	3	
Name	Handle Transaction	
Summary	Verifying Customer transaction	
Priority	4	
Pre-Condition	1. Log into the system. 2. System should be up and running.	
Post-Condition	Handling Transaction Properly.	
Primary Actor	Manager	
Trigger	Manager can access transaction details and verify transaction	
Main Scenario	**Step**	**Action**
	1	System displays Login page.
	2	Manager enter username and password.
	3	System validates login details.
	4	System displays successful message and menu.
	5	Manager clicks on transaction details.
	6	Manager clicks on pay salaries.
	7	System displays a list of transaction details.
	8	System displays salaries.
	9	System validates transactions.
	10	System displays “Validation is successfully”.
	11	Display logout button and need a receipt button.
	12	Manager logout from the system.
Extension	**Step**	**Branching Action**
	3.a	If Manager name and password are incorrect. 3.a.1 – System shows credentials are invalid. 3.a.2 – System indicates a prompt box to re-enter valid credentials.
	11.a	If the amount Manager need to pay salaries is greater than account balance. 11.a.1 – System displays error message. 11.a.2 – System displays home page.
	12.a	If Manager need a receipt. 12.a.1 – System prints and give a receipt to the Manager. 12.b.2 – Manager logout from the system.

Sequence Diagram




Handle Transaction
Manager
<<Boundry>>
Login UI : Class1
<<control>>
Controller : Class2
<<Entity>>
Customer : Class3
<<entity>>
Print : Class4
ref
Login
loop
until details are valid
opt
[if password and username is incorrect]
DisplayErrorMessage()
DisplayLoginPage()
ClicksOnTransactionDetails(1)
Transaction(1)
Transaction(1)
DisplayListOfTransactionDettails()
ClicksOnPaySalaries(a)
PaySalaries(a)
PaySalaries(a)
DisplayPaySalaries()
ValidateTransaction(2)
Transaction(2)
Transaction(2)
par
DisplayValidationSuccessfullMessage()
DisplayLogoutAndNeedAReciptButtons()
opt
[if salary > account balance]
DisplayErrorMessage()
DisplayHomePage()
receipt
[if Manager need a receipt]
RequestForReceipt()
RequestForReceipt()
RequestForReceipt()
receipt
receipt
Logout
Logout
Logout

Communication Diagram

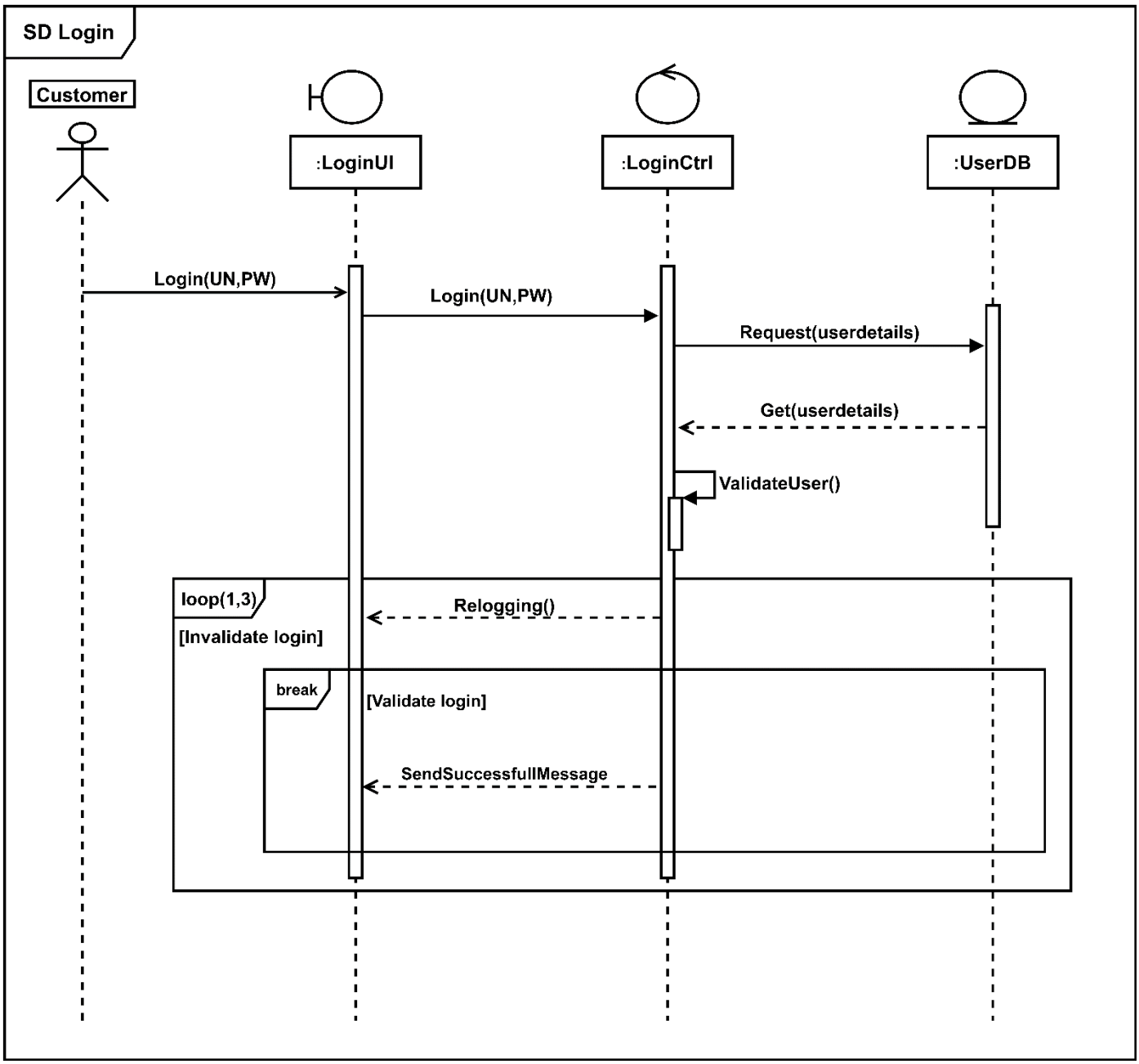


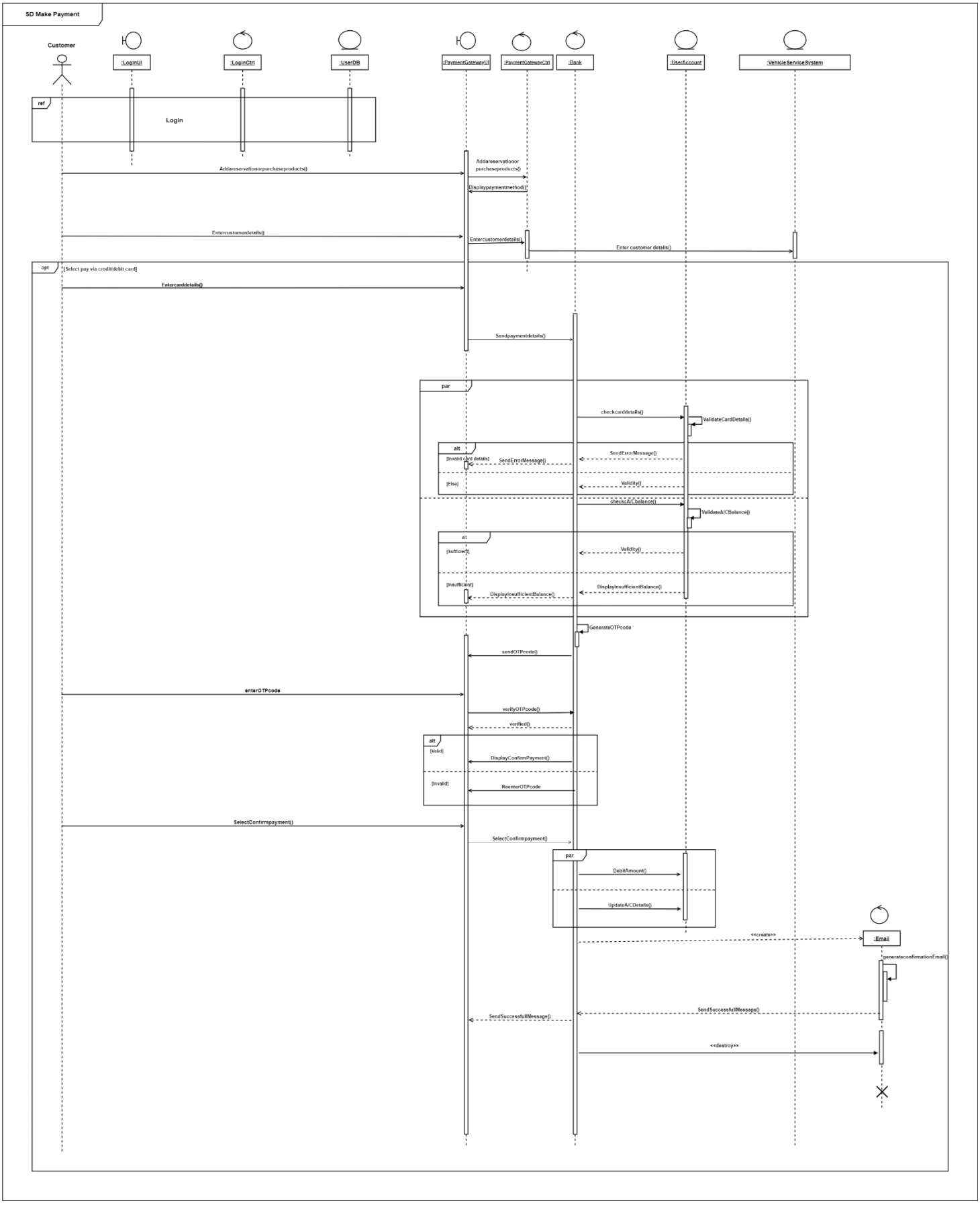
Use Case Scenario

Number	4	
Name	Make payment	
Summary	Customer has made the payment for services, items and reservations	
Priority	**2**	
Pre-conditions	Customer has logged to the system.	
Post-conditions	Customer has successfully made the payment.	
Primary Actor	Customer	
Trigger	Customer has chosen to make payment for services, items and reservations.	
Main Scenario	**Step**	**Action**
	01	Customer logs into the system.
	02	Customer enter username and password.
	03	Customer adds a reservation or purchase products.
	04	Payment gateway displays payment method pay via Credit/ Debit card or pay on cash.
	05	Customer selects payment method and enters details.
	06	Payment gateway sends payment details to the bank.
	07	Bank checks if details are valid and if user's account balance is sufficient for transaction at the same time by validating.
	08	The bank creates a onetime password (OTP) and transmits it to the customer via the payment gateway.
	09	Customer enter OTP code into the payment gateway.
	10	Payment gateway verifies OTP code with bank generated OTP code.
	11	Payment gateway displays "Confirm payment".
	12	Customer selects confirm payment.
	13	The amount is deducted from the account of the user.
	14	Customer account details are updated.
	15	Bank creates payment confirmation email.

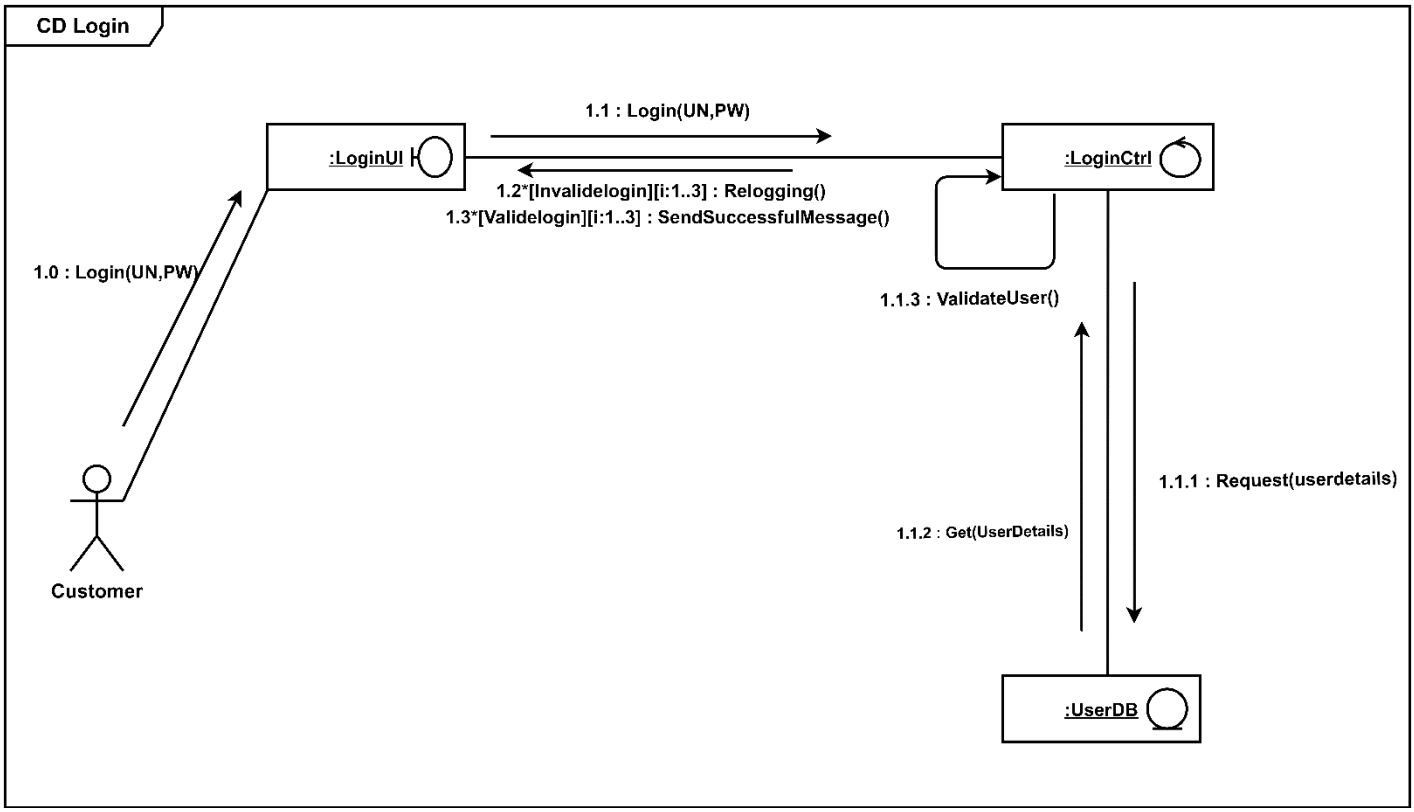
	16	Bank sends the email to the customer.
Extensions	**Step**	**Branching Action**
	02.a	If customer enter incorrect username or password, system displays incorrect username or password please re login.
	02.b	After three times, system displays try again in a little while.
	05.a	If customer select pay on cash, customer should enter customer details.
	05.b	If customer selects pay via Credit/Debit card, customer should enter card details.
	07.a	If the customer does not provide valid card number, bank will provide the message reenter card number again.
	07.b	If customer account balance is insufficient, customer is directed to select payment method and enter details page and insufficient balance is displayed.
	10.a	If OTP code does not verify, customer is directed to reenter OTP code.

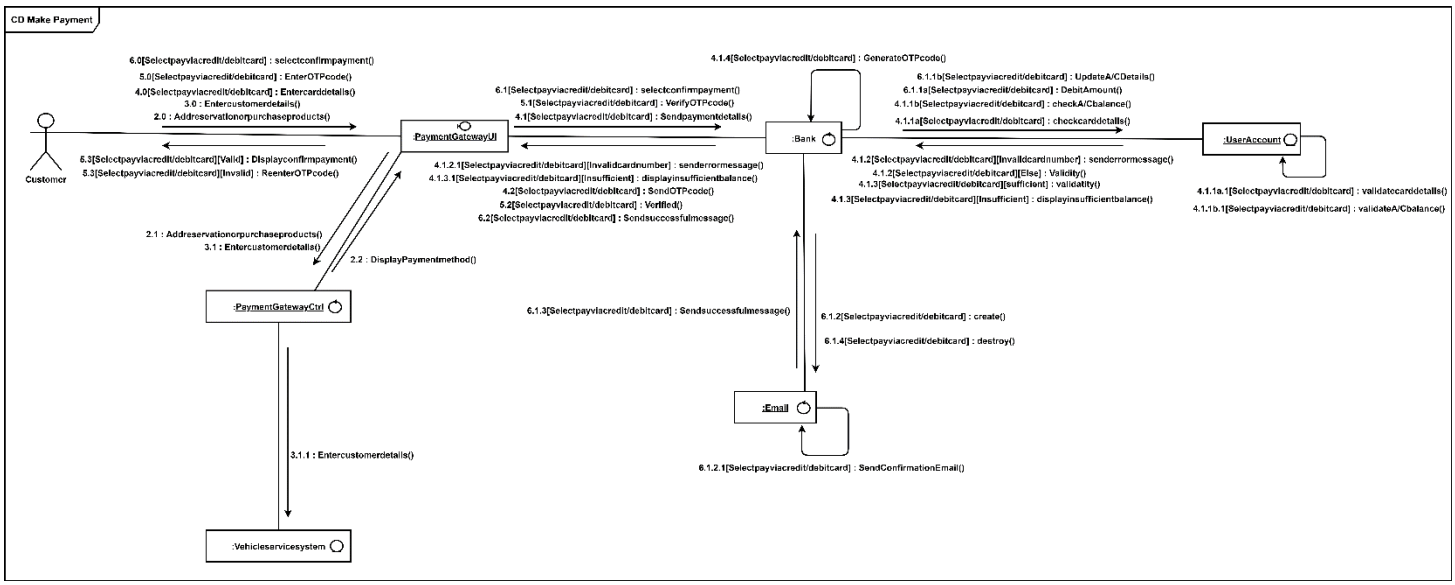
Sequence Diagram




SD Make Payment
Customer
:LoginUI
:LoginCtrl
:UserDB
:PaymentGatewayUI
:PaymentGatewayCtrl
:Bank
:UserAccount
:VehicleServiceSystem
ref
Login
Addareservationorpurchaseproducts()
Addareservationor purchaseproducts()
Displaypaymentmethod()
Entercustomerdetails()
Entercustomerdetails()
Enter customer details()
opt
[Select pay via credit/debit card]
Entercarddetails()
Sendpaymentdetails()
par
checkcarddetails()
ValidateCardDetails()
alt
[Invalid card details]
SendErrorMessage()
SendErrorMessage()
[Else]
Validity()
checkA/Cbalance()
ValidateA/CBalance()
alt
[Sufficient]
Validity()
[Insufficient]
DisplayInsufficientBalance()
DisplayInsufficientBalance()
GenerateOTPcode
sendOTPcode()
enterOTPcode
verifyOTPcode()
verified()
alt
[Valid]
DisplayConfirmPayment()
[Invalid]
ReenterOTPcode
SelectConfirmpayment()
SelectConfirmpayment()
par
DebitAmount()
UpdateA/CDetails()
<<create>>
:Email
generateconfirmationEmail()
SendSuccessfullMessage()
SendSuccessfullMessage()
<<destroy>>

Communication Diagram




CD Make Payment
6.0[Selectpayviacredit/debitcard] : selectconfirmpayment()
5.0[Selectpayviacredit/debitcard] : EnterOTPcode()
4.0[Selectpayviacredit/debitcard] : Entercarddetails()
3.0 : Entercustomerdetails()
2.0 : Addreservationorpurchaseproducts()
Customer
5.3[Selectpayviacredit/debitcard][Valid] : Displayconfirmpayment()
5.3[Selectpayviacredit/debitcard][Invalid] : ReenterOTPcode()
:PaymentGatewayUI
6.1[Selectpayviacredit/debitcard] : selectconfirmpayment()
5.1[Selectpayviacredit/debitcard] : VerifyOTPcode()
4.1[Selectpayviacredit/debitcard] : Sendpaymentdetails()
4.1.2.1[Selectpayviacredit/debitcard][Invalidcardnumber] : senderrormessage()
4.1.3.1[Selectpayviacredit/debitcard][Insufficient] : displayinsufficientbalance()
4.2[Selectpayviacredit/debitcard] : SendOTPcode()
5.2[Selectpayviacredit/debitcard] : Verified()
6.2[Selectpayviacredit/debitcard] : Sendsuccessfulmessage()
2.1 : Addreservationorpurchaseproducts()
3.1 : Entercustomerdetails()
2.2 : DisplayPaymentmethod()
:PaymentGatewayCtrl
3.1.1 : Entercustomerdetails()
:Vehicleservicesystem
4.1.4[Selectpayviacredit/debitcard] : GenerateOTPcode()
:Bank
6.1.1b[Selectpayviacredit/debitcard] : UpdateA/CDetails()
6.1.1a[Selectpayviacredit/debitcard] : DebitAmount()
4.1.1b[Selectpayviacredit/debitcard] : checkA/Cbalance()
4.1.1a[Selectpayviacredit/debitcard] : checkcarddetails()
4.1.2[Selectpayviacredit/debitcard][Invalidcardnumber] : senderrormessage()
4.1.2[Selectpayviacredit/debitcard][Else] : Validity()
4.1.3[Selectpayviacredit/debitcard][sufficient] : validatity()
4.1.3[Selectpayviacredit/debitcard][Insufficient] : displayinsufficientbalance()
:UserAccount
4.1.1a.1[Selectpayviacredit/debitcard] : validatecarddetails()
4.1.1b.1[Selectpayviacredit/debitcard] : validateA/Cbalance()
6.1.3[Selectpayviacredit/debitcard] : Sendsuccessfulmessage()
6.1.2[Selectpayviacredit/debitcard] : create()
6.1.4[Selectpayviacredit/debitcard] : destroy()
:Email
6.1.2.1[Selectpayviacredit/debitcard] : SendConfirmationEmail()

Individual contribution

Reg.No	Name	Contribution
IT20625016	Nanayakkara H.M.S.Y.	1.**Use case scenario** (Find nearest filling station) 2.**Sequence diagram** i. Reference sd- customer login ii. Sd – Find nearest filling station 3.**Communication diagrams** i. cd – Find nearest filling station
IT20606060	Wanni Arachchige H.S.	1.**Use case scenario** (Update the system) 2.**Sequence diagram** i. Reference sd - Login to the admin dashboard ii. Sd – Update the System 3.**Communication diagrams** i. cd – Update the System
IT20297640	Piyasinghe H.K.N.Y.	1.**Use case scenario** (Handle transactions) 2.**Sequence diagram** i. Reference sd- Manager login ii. Sd – Handle transactions 3.**Communication diagrams** i. cd – Handle transactions
IT20150648	Jayakody J.A.M.G.	1.**Use case scenario** (Make payment) 2.**Sequence diagram** i. Reference sd- customer login ii. Sd – Make payment 3.**Communication diagrams** i. cd – Customer login ii. cd- Make payment