HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

SOFTWARE ENGINEERING

Project Submission 4

Lecturer: Quan Thanh Tho
Author: Vuong Le Huy

4/2020

Table 1: GROUP'S MEMBERS

No.	Name	ID
1	Trinh Mai Duy	1752139
2	Truong Van Quang Dat	1652142
3	Nguyen Tri Duc	1652160
4	Vuong Le Huy	1652252

Table 2: ACTIVITY LOG

Day	Changes	Member
21/04/2020	Non-Functional Requirement; Add Document's History; Add Group's Members	Trinh Mai Duy
21/04/2020	Add Introduction; Add 3 non-functional requirements	Truong Van Quang Dat
22/04/2020	Add use-case diagram for the whole system	Trinh Mai Duy
23/04/2020	Check and fix use-case diagram for the whole system; Add header and footer. Add a non-functional requirement;	Vuong Le Huy
23/04/2020	Check overall project; Add a non-functional requirement; Add cover and some minus details.	Nguyen Tri Duc

Table 3: CHANGE LOG

Version	Changes	Section	Page
Submission 01	Non-Functional Requirements Non-interactive Functional Requirement(Bonus) Use-case Scenario	5.3 5.2 7	5 5 10;11;12;13
Submission 02	Sequence Diagram	9	17
Submission 03	Implementation View	11	20

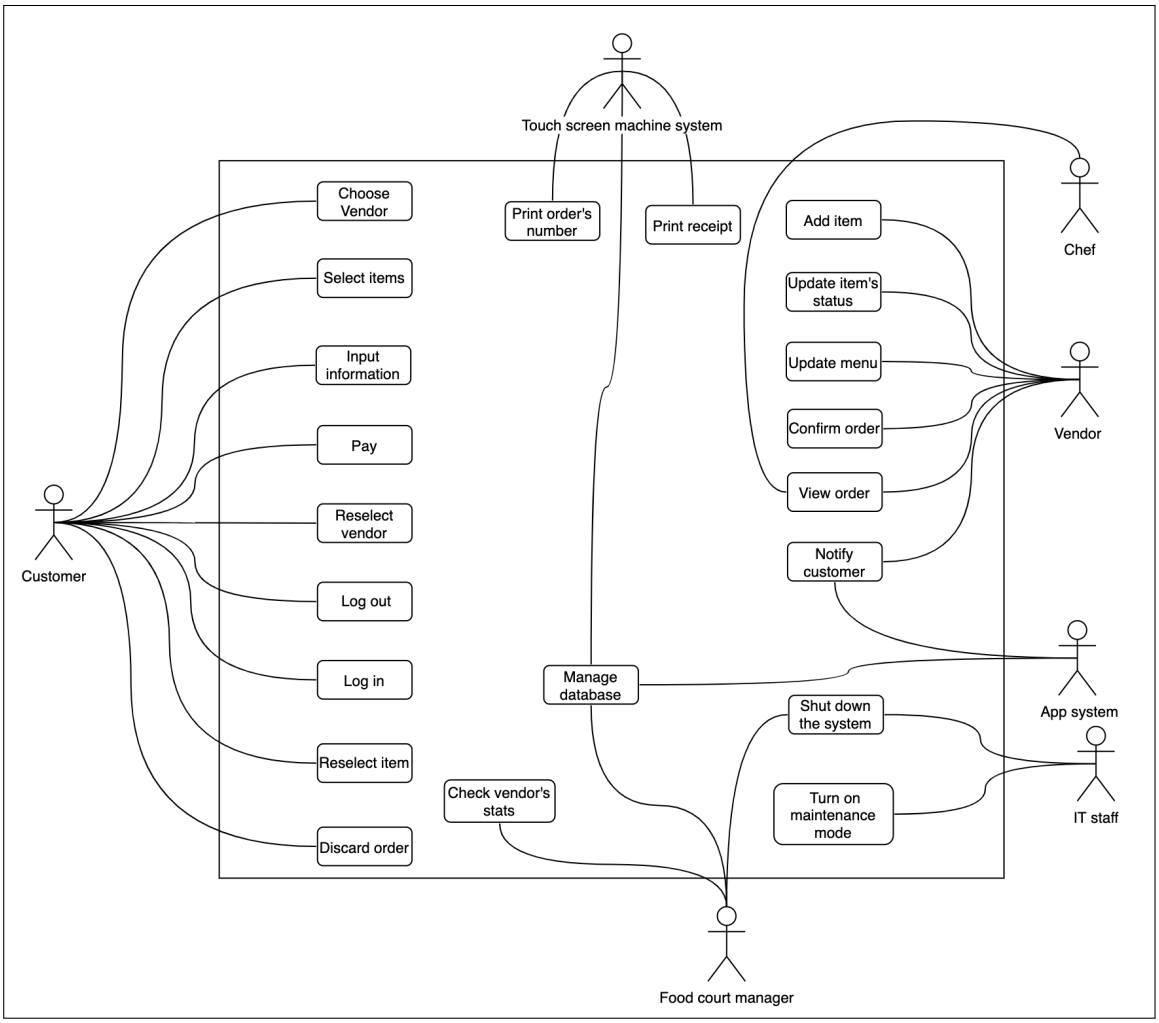
Contents

1 Introduction

Nowadays, the food court of our university has so many guests at a same time especially lunch time. Therefore the food court can not control every activities well and it will make the guests fell uncomfortable which can decrease profit of the vendors.

In our project, we create the system which can control and manage many activities of food court in our university. It includes payment and ordering that can be done online to save the time of the guests.

2 Use-case Diagram for the system



3 Primary Project's Requirements

- It specifies the external system behaviors.
- It specifies constraints on the implementation.
- It is easy to change.
- It serves as reference tool for system maintainers.
- It records forethought about the life cycle of the system
- It characterizes acceptable response to understand events.

4 Performance Requirements

In order to maintain an acceptable speed at maximum number of uploads allowed from a particular customer will be any number of users can access the system at anytime.

Also connections to the servers will be based on the criteria of attributes of the user like his location and server should stably operates in working time.

5 Universal System's Functional & Non-Functional Requirements

5.1 Functional Requirements

- Keep records of admission of customers.
- Keep daily sell.
- Store feedback given by customers.
- keep details about delivered product.

5.2 General non-functional requirements for the system

- The app must be compatible on Android and IOS.
- Customer will be able to pay through a wide variety of options (MoMo wallet, Zalo pay,QR code, directly with ATM card or VISA card,.....).
- The system will be available as long as the Food Court is open.
- Work with Banks, Online Wallet so that every time user make a purchase, an OTP codewill be sent to user to confirm and finalize the transaction.
- With user using the app, OTP code will only be required for the linking card to user's account. After linking the Debit/Credit card to the account, user can proceed to payment using fingerprint (if user's phone support) or a short PIN code set by user.

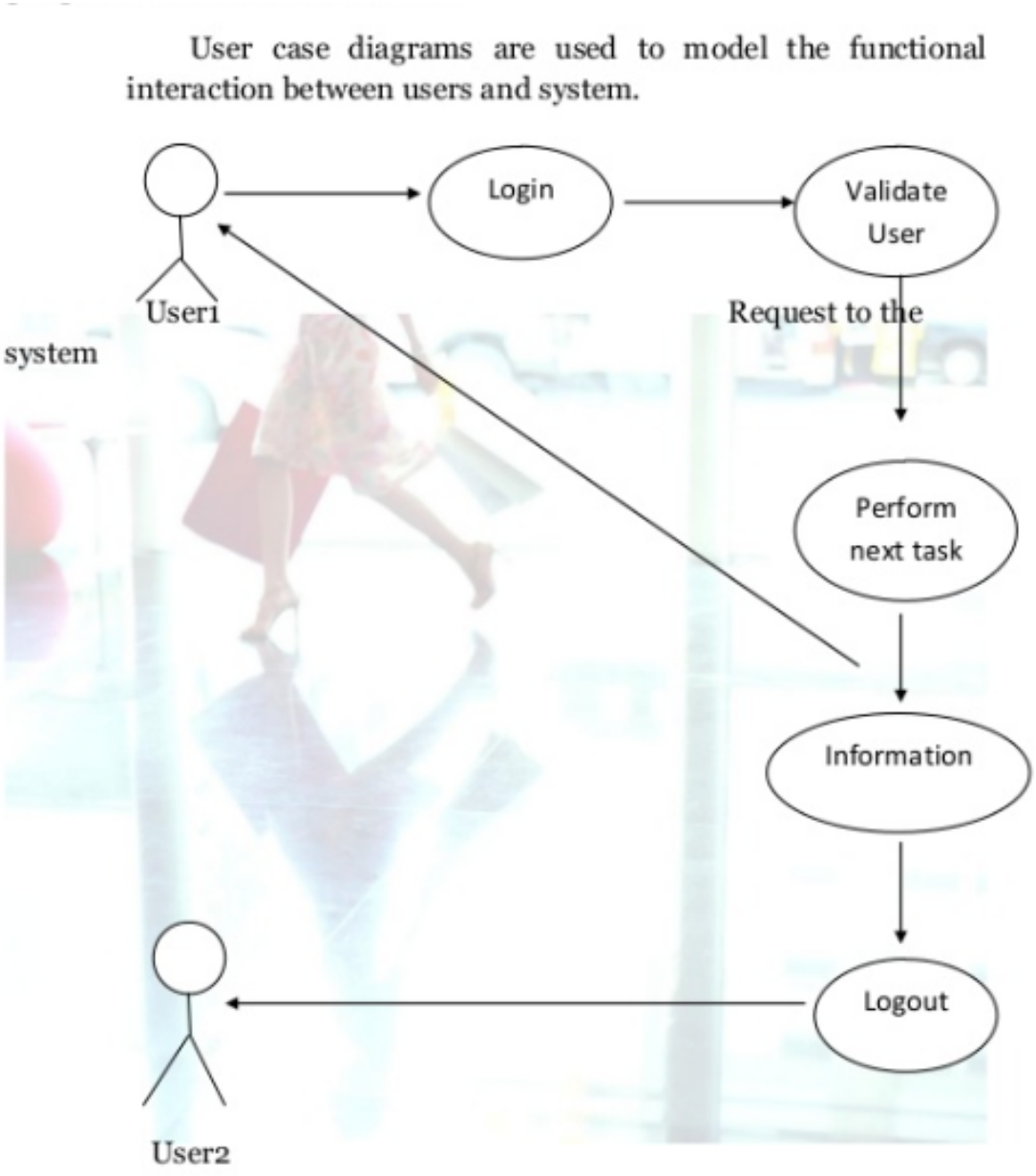
5.3 Non-Functional Requirements (Individual)

- The app must be compatible on Android and IOS.
- Customer will be able to pay through a wide variety of options (MoMo wallet, Zalo pay, QR code, directly with ATM card or VISA card,.....).
- The system will be available as long as the Food Court is open.
- Work with Banks, Online Wallet so that every time user make a purchase, an OTP code will be sent to user to confirm and finalize the transaction.
- With user using the app, OTP code will only be required for the linking card to user's account. After linking the Debit/Credit card to the account, user can proceed to payment using fingerprint (if user's phone support) or a short PIN code set by user.
- The mobile app shall take no more than 200MB of storage and generate at most 50MB of cached data.
- App update is mandatory if the current version is 1 month out of date.
- Food ordering on mobile app is only available to anyone with an active hcmut.edu.vn email. account.
- The order from the time the customer press finish on the screen until it reach the system should not be more than 10 seconds.

5.4 Non-interactive Functional Requirement (Bonus)

- System shall reboot when memory is 100% occupied.
- System shall record the amount of active user and send to the manager.
  - System shall store data and analyse that data every month in order to evaluate how the system works. This does not interact with any system, it is non-interactive.
- System shall record complaint of customer's request and send to the manager.
  - By receiving calls or through emails. The shortcoming of the system will be pointed out and fixed. This also is a non-interactive function.
- System records the presence of the employees by the fingerprint.
  - This system will help the employer to know the working efficient of the food court.

6 User Case Tabular & Diagram



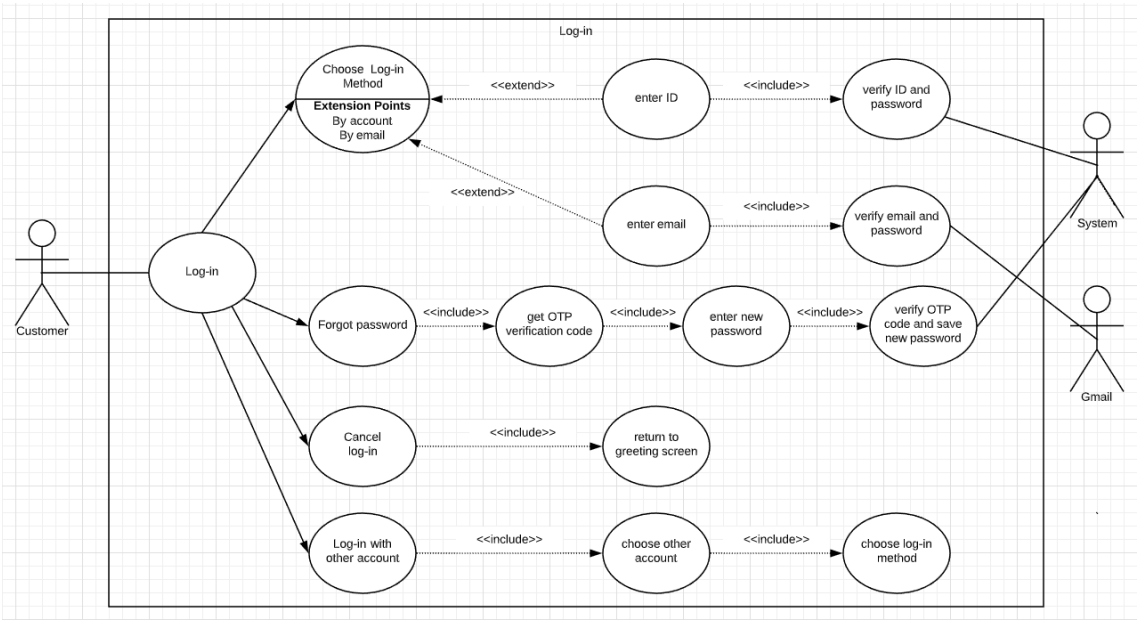
(User Case Diagram)

Use Case	Description
Register	A new CUSTOMER needs to first register into the system before performing any transaction. Actor/s: CUSTOMER Pre-condition: An unregistered CUSTOMER. Main flow of events: 1. The CUSTOMER clicks the REGISTER button on the Home Page. 2. The system displays the Register Page. 3. The CUSTOMER enters all of the required information. 4. The CUSTOMER clicks the SEND button. 5. The system checks that all of the required information were entered. If yes, the system update the CUSTOMER's record in the CUSTOMER and ACCOUNT tables in the database. System displays OK message. Post-condition: The new CUSTOMER has registered. The ACCOUNT and CUSTOMER tables are updated.
Log-in	A CUSTOMER needs to log-in before performing any transaction Actor/s: CUSTOMER Pre-condition: A registered user. Main flow of events: 1. The CUSTOMER clicks the Log-in button on the Home Page.

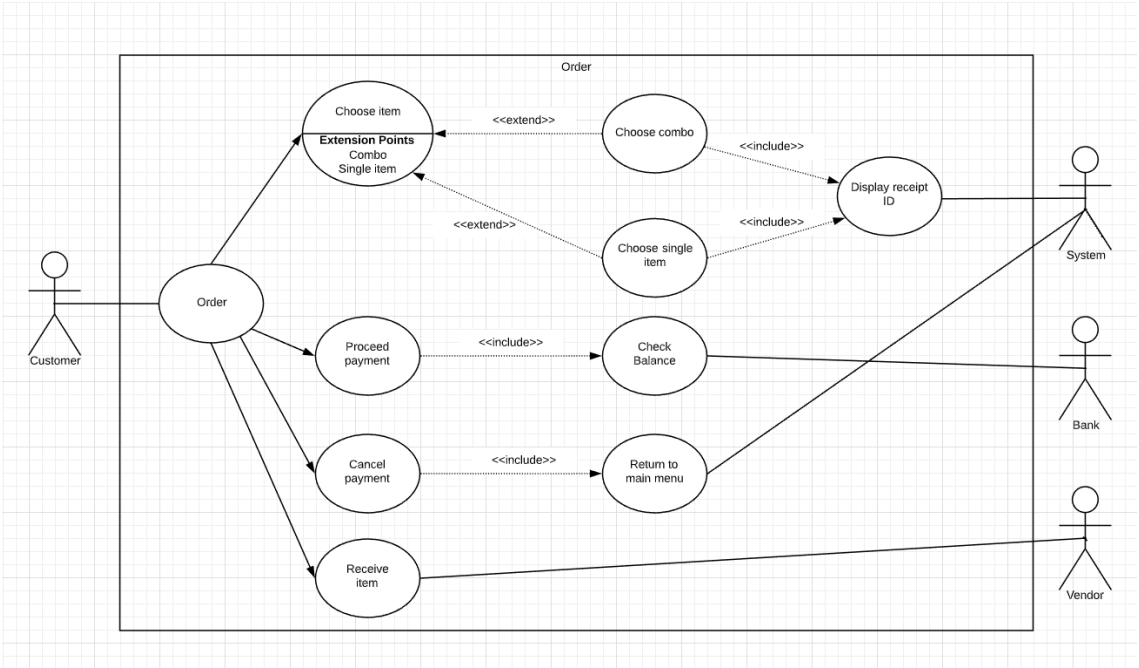
Use Case	Description
Log-in (continue …)	2. The system displays the Log-in Page. 3. The CUSTOMER enters his/her user ID and password. 4. The CUSTOMER clicks the OK button. 5. The system validates the log-in information against the ACCOUNT table in the database. 6. CUSTOMER is an authorised user; the system displays the Personal Home Page to the CUSTOMER Post-condition: The CUSTOMER has been authorised to perform transactions. Alternate flow: 1. The CUSTOMER clicks the Log-in button on the Home Page. 2. The system displays the Log-in Page. 3. The CUSTOMER enters his/her user ID and password. 4. The CUSTOMER clicks the OK button. 5. The system validates the log-in information against the ACCOUNT table in the database. 6. CUSTOMER is not an authorised user; the system displays a pop-up message to inform the CUSTOMER.

Use Case	Description
Check Out	1. The system displays the books in the ORDER table of the CUSTOMER on the web Page. 2. The CUSTOMER checks the order list for any inconsistency. If nothing found, CUSTOMER clicks the PROCEED button. 3. The system displays the Invoice page. 4. The Customer enters the relevant credit card information and clicks the OK button. 5. The system checks that the credit card is valid. Then, the system displays the Delivery Details page. 6. The CUSTOMER chooses destination for delivery, along with delivery options. Then, he/she clicks the PROCEED button. 7. The system will display the check-out information for confirmation. 8. The CUSTOMER checks that all information is correct and then

7 Diagram for A Specific Use-case Scenario



Use Case Diagram for Log-in on App



Use Case Diagram for Order on App

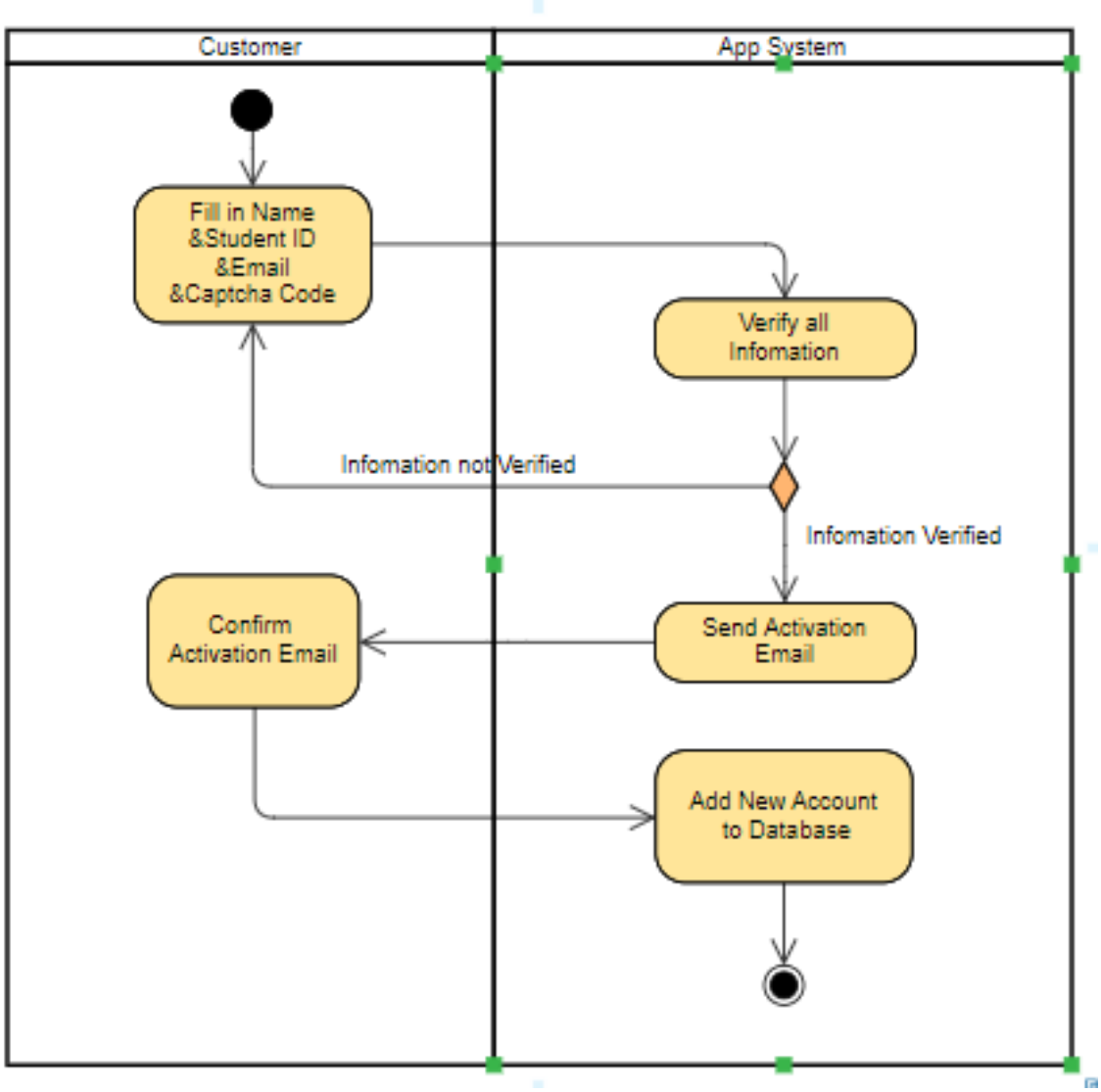
Use Case name	**Register**
Use Case ID	UC-1.1
Description	Customer wants to register to begin to use the service.
Actor	Customer.
Priority	Must have.
Trigger	Customer wants to register to the app.
Pre-condition	Customer already has @hcmut account. Customer's computer has access to the internet.
Post-condition	System creates a new account.
Basic flow	1. Customer clicks the register button on Home page. 2. System displays Register page. 3. Customer enters name, student ID and mobile phone number. 4. Customer clicks send button. 5. System sends OTP to the phone number. 6. System verifies information. 7. System accesses to database. 8. System adds new account information to database. 9. System displays register success message.
Alternative flow	8.1 System recognizes account is already added to database beforehand. 8.2 System displays account already created message. Use case stops
Exception flow	6.1. System fails to verify and displays message. 6.1.1 Customer chooses cancel Use case stops

Use Case name	**Log-in**
Use Case ID	UC-1.2
Description	User wants to log in to use the service.
Actor	Customer, admin, vendor.
Priority	Must have.
Trigger	User wants to log in to the app.
Pre-condition	User already has a account. User's computer has access to the internet.
Basic flow	1. User opens the app. 2. User chooses log in options. 3. User enters ID and password and tap Log-in button. 4. System verifies information and grant access. 5. System writes log-in success to Activity log.
Alternative flow	2.1 User chooses log in with g-mail. 2.1.1 System opens g-mail log in page. 3.1 User enters google ID and password and log in. 4.1 Google verifies log in and grant access to account. Use case continues on step 5.
Exception flow	4.2. System fails to verify and displays message. 4.2.1 User chooses cancel Use case stops 4.2.2 User chooses forgot password. Use case continues with Use Case UC-Recover account.
Business rules	BR1.1-1: System locks account if user enter incorrect password five times.
NFR	NFR1.1-1: Password shall be encrypted using hash function MD5.

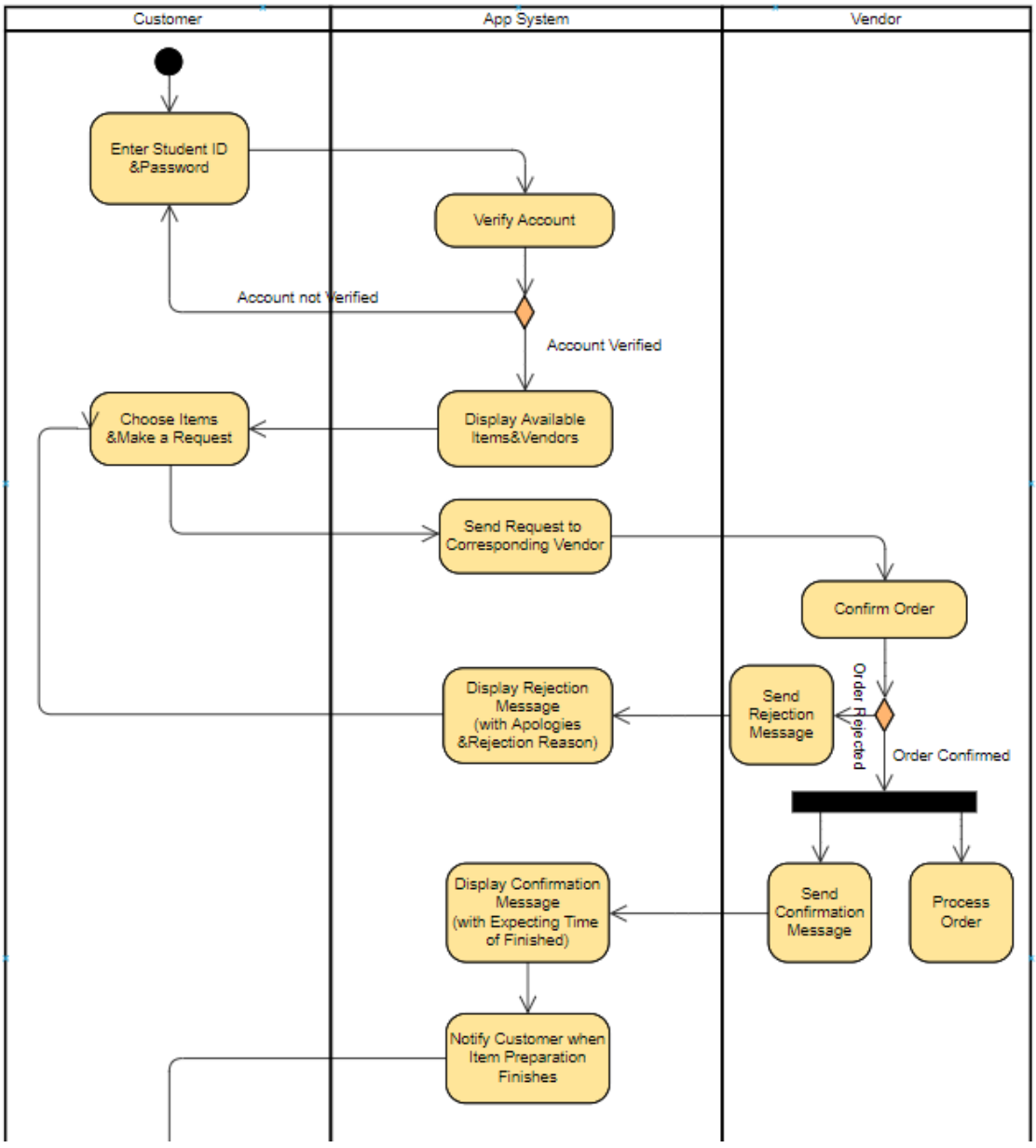
Use Case name	**Order**
Use Case ID	UC-1.3
Description	Customer wants to order items.
Actor	Customer.
Priority	Optional.
Trigger	Customer wants to order items by app.
Pre-condition	Customer already has a account. Customer already installed app on his/her device. Customer's device has access to the internet.
Basic flow	1. Customer opens the app. 2. Customer logs in the system. 3. Customer chooses items. 4. System sends request to vendor. 5. Vendor confirms request. 6. System displays payment page. 7. Customer choose payment method. 8. System displays receipt ID. 9. Vendor delivers item . 10. Vendor sends item delivered message to system. 11. System displays item delivered message. 12. System displays main menu.
Alternative flow	3.1 Customer clicks combo button. 3.1.1 System displays combo menu. Use case continues on step 4. 3.2 Customer clicks single item button. 3.2.1 System displays item menu. Use case continues on step 4. At step 3, 7, if user choose to cancel order, system returns to the main menu. At step 7 if the user want more food or drinks it will return to step 2.
Exception flow	7.1 Balance is insufficient. 7.2 System displays insufficient balance message. 7.2.1 Customer clicks cancel. Use case stops. 7.2.2 Customer clicks top-up. Use case continues with UC-top-up

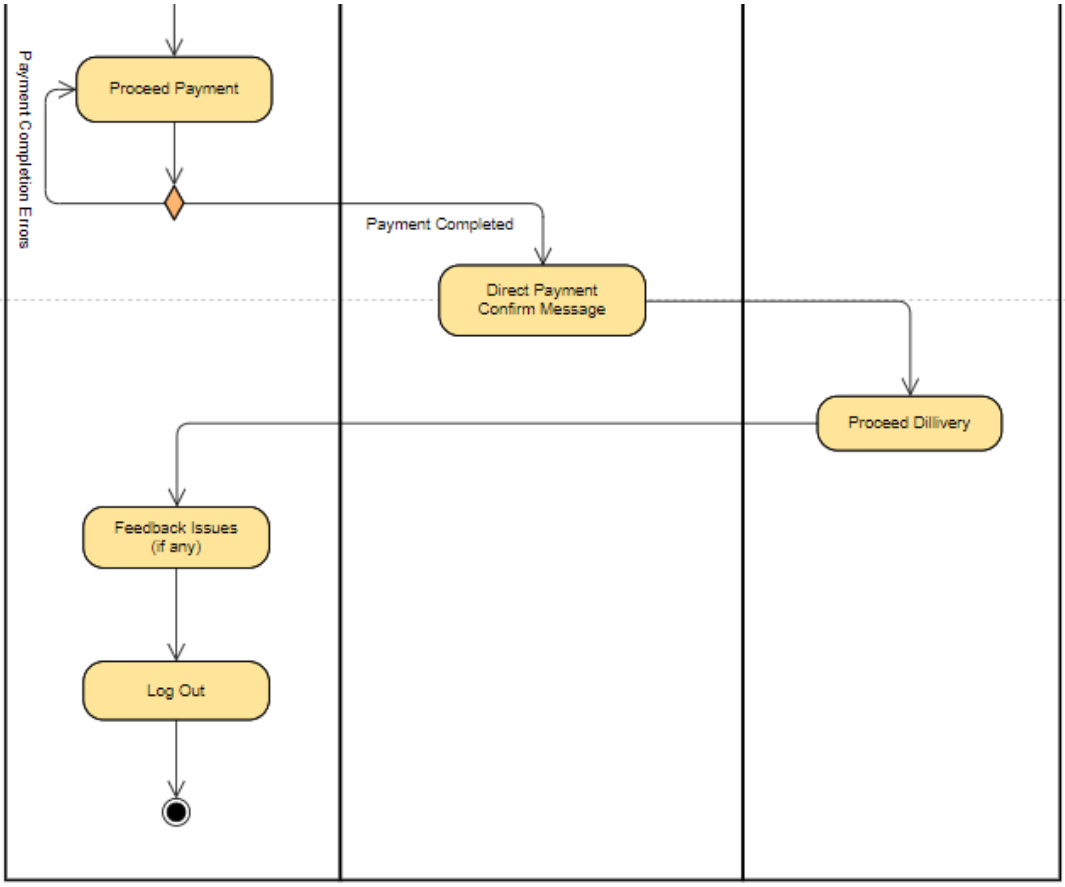
8 Activity Diagrams

8.1 Customer Registration

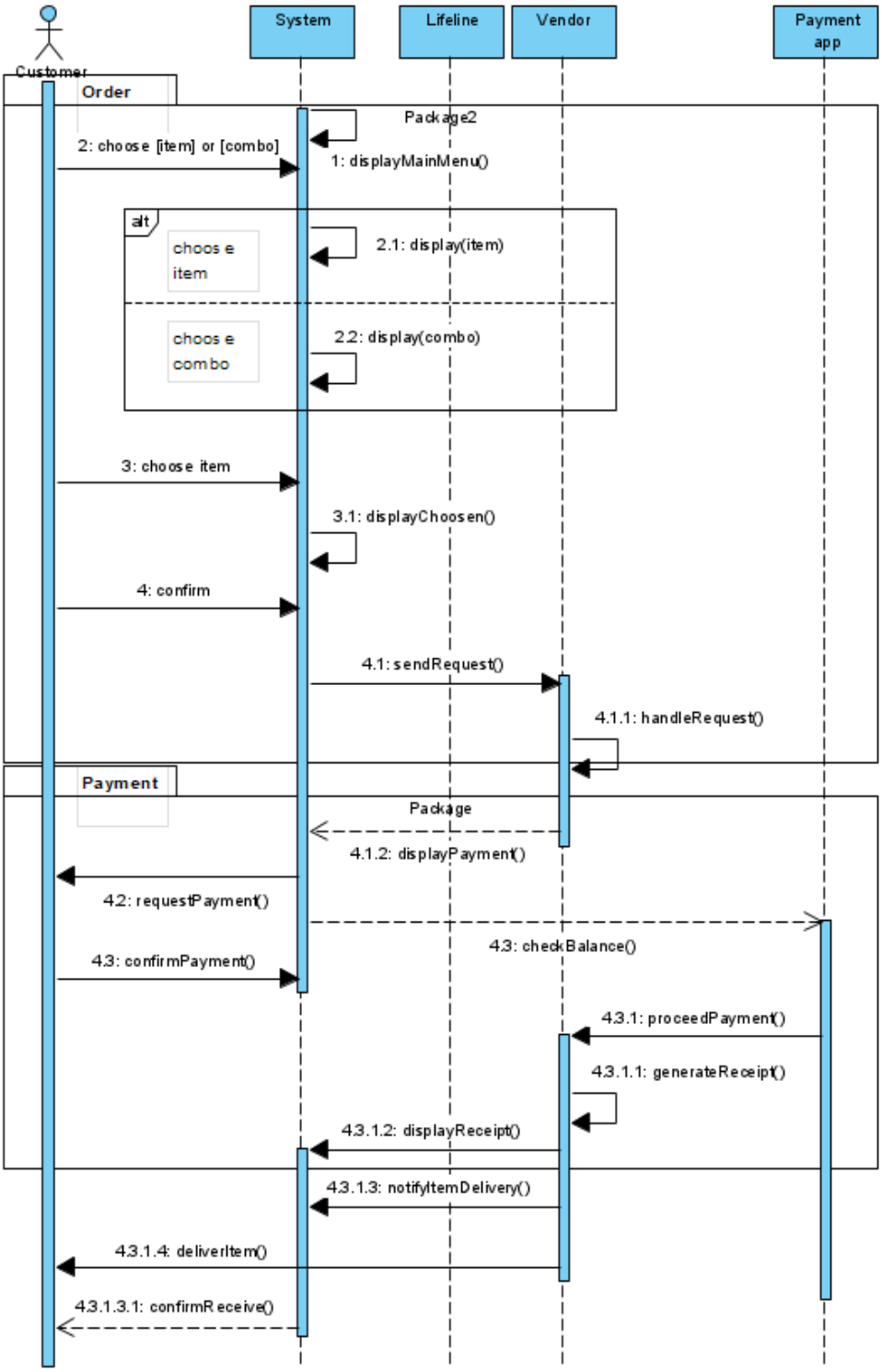


8.2 Customer Ordering

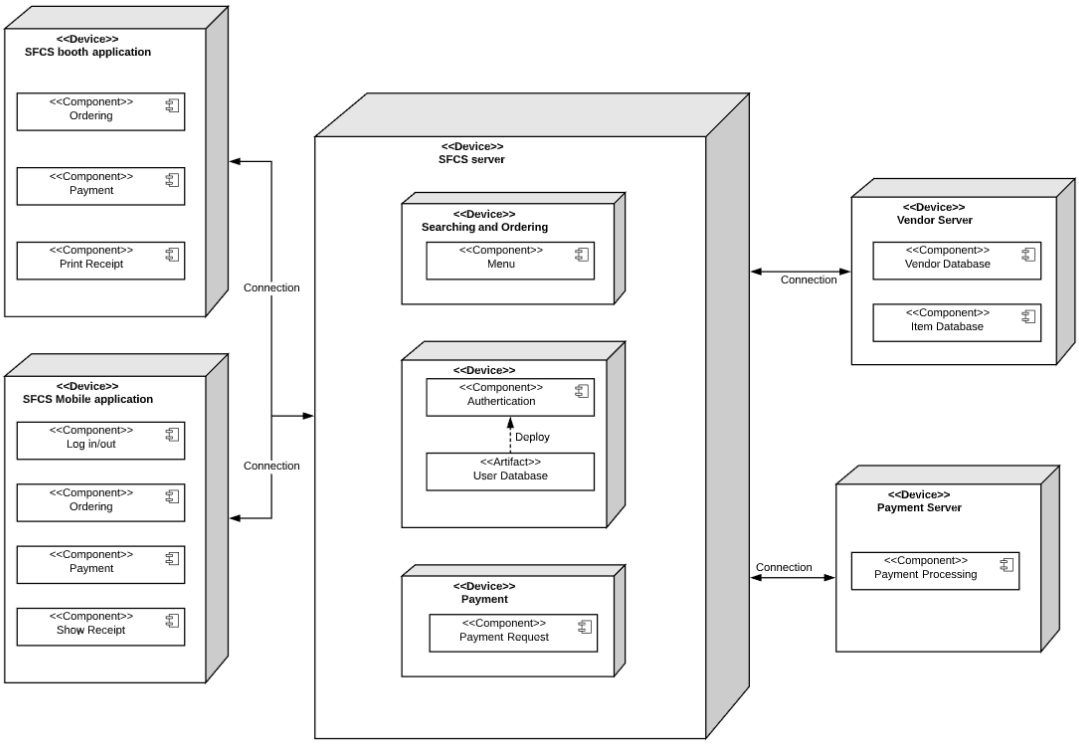


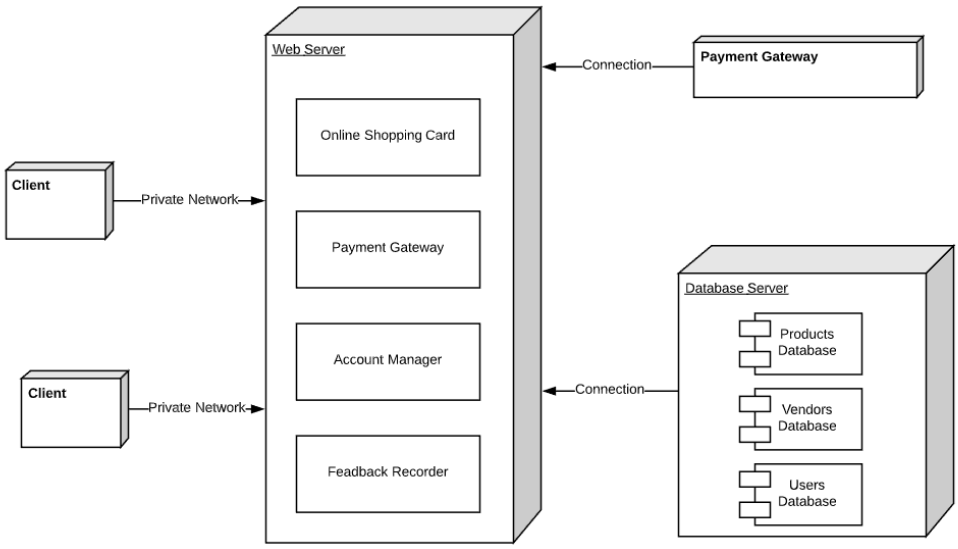

Payment Completion Errors
Proceed Payment
Payment Completed
Direct Payment
Confirm Message
Proceed Dillivery
Feedback Issues
(if any)
Log Out

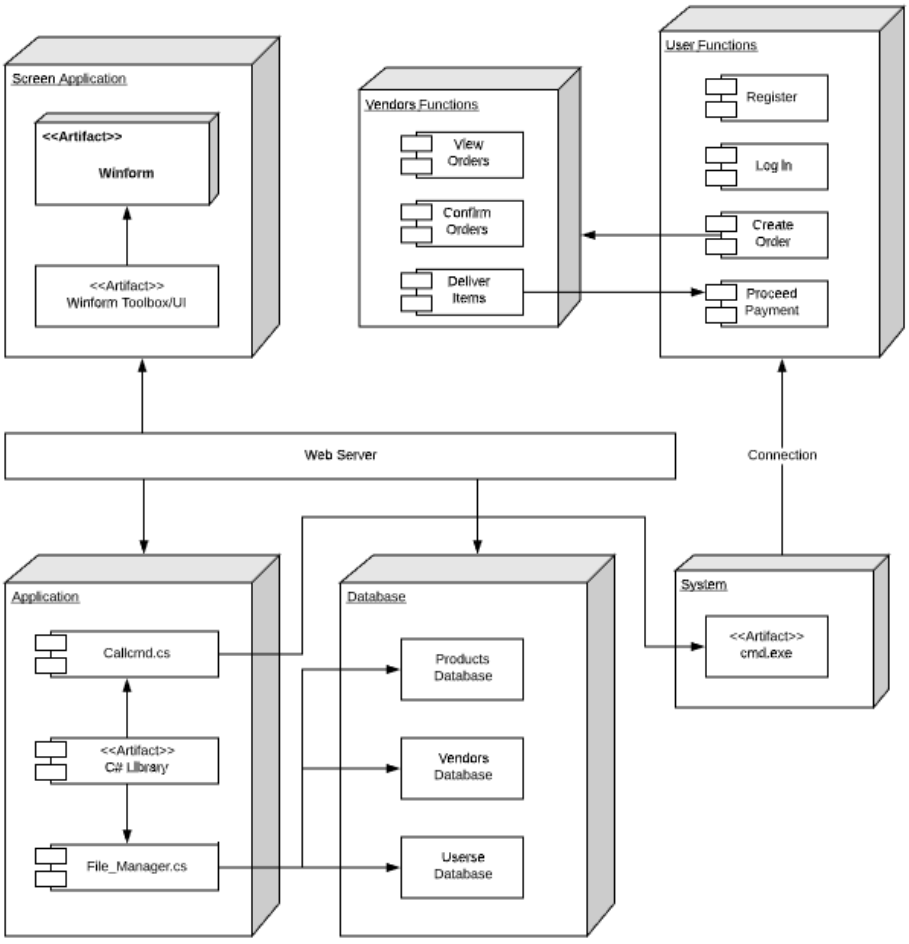
9 Sequence Diagram



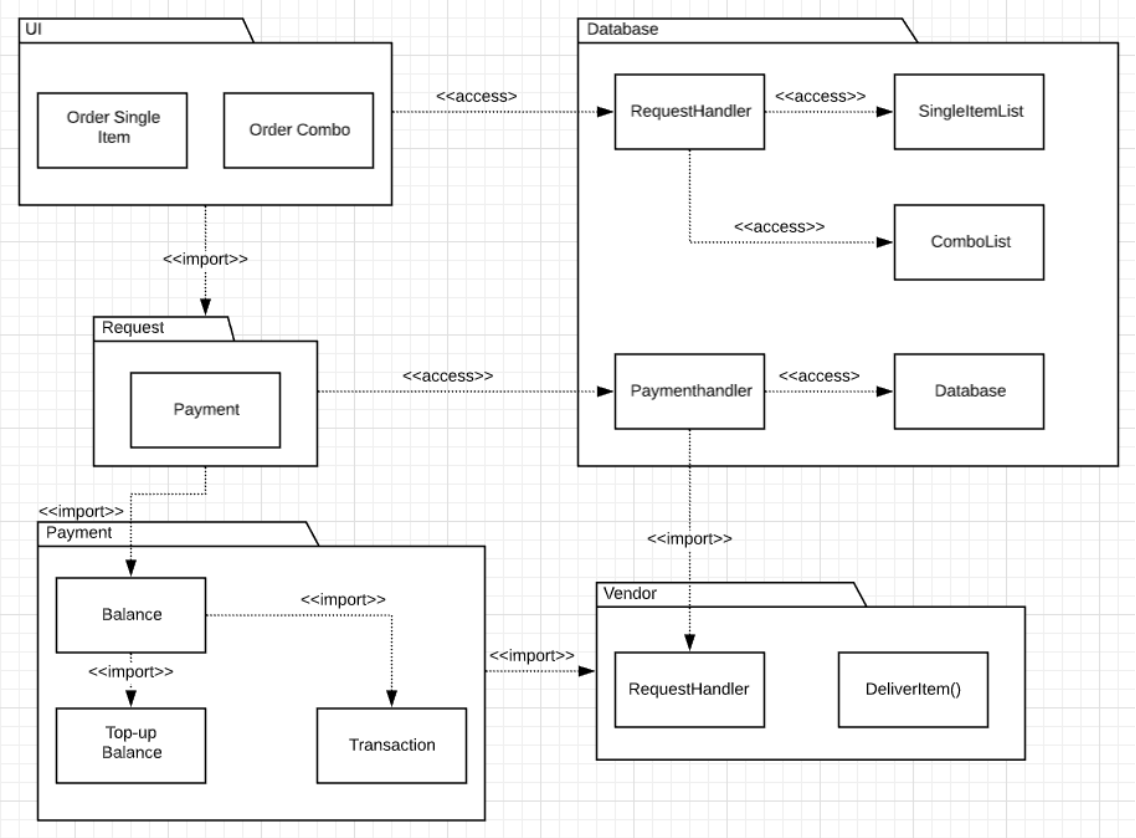
10 Deployment View



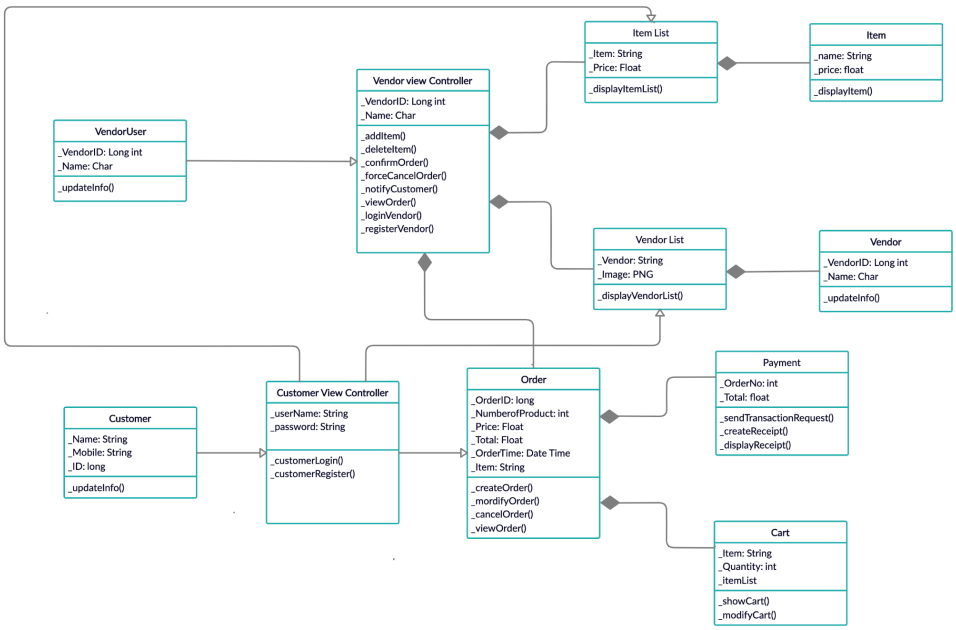



Screen Application
<<Artifact>>
Winform
<<Artifact>>
Winform Toolbox/UI
Vendors Functions
View Orders
Confirm Orders
Deliver Items
User Functions
Register
Log In
Create Order
Proceed Payment
Web Server
Connection
Application
Callcmd.cs
<<Artifact>>
C# Library
File_Manager.cs
Database
Products Database
Vendors Database
Userse Database
System
<<Artifact>>
cmd.exe

11 Implementation View



12 Class Diagram



13 Module List

Vendor view controller:

- additem(): add more item
- deleteitem(): remove item
- confirmOrder(): confirm the order

  forceCancelOrder(): cancel the order notifyCustomer(): send notification to customer viewOrder(): display the order loginVendor(): vendor login in to the app registerVendor(): regist new vendor to the system

Item list:

- displayitemList(): display the item list

Vendor list:

- displayvendorList(): display the vendor list

Vendor User:

- updateinfo(): update information

Item list:

- createOrder(): create new order
- modifyOrder(): change the order
- cancelOrder(): cancel the order
- viewOrder(): display the order

Customer View Controller:

- customerLogin(): customer login into the system
- customerRegister(): customer create an account

Customer:

- updateinfo(): update information

Item:

- displayitem(): display the item

Vendor:

- updateinfo(): update information

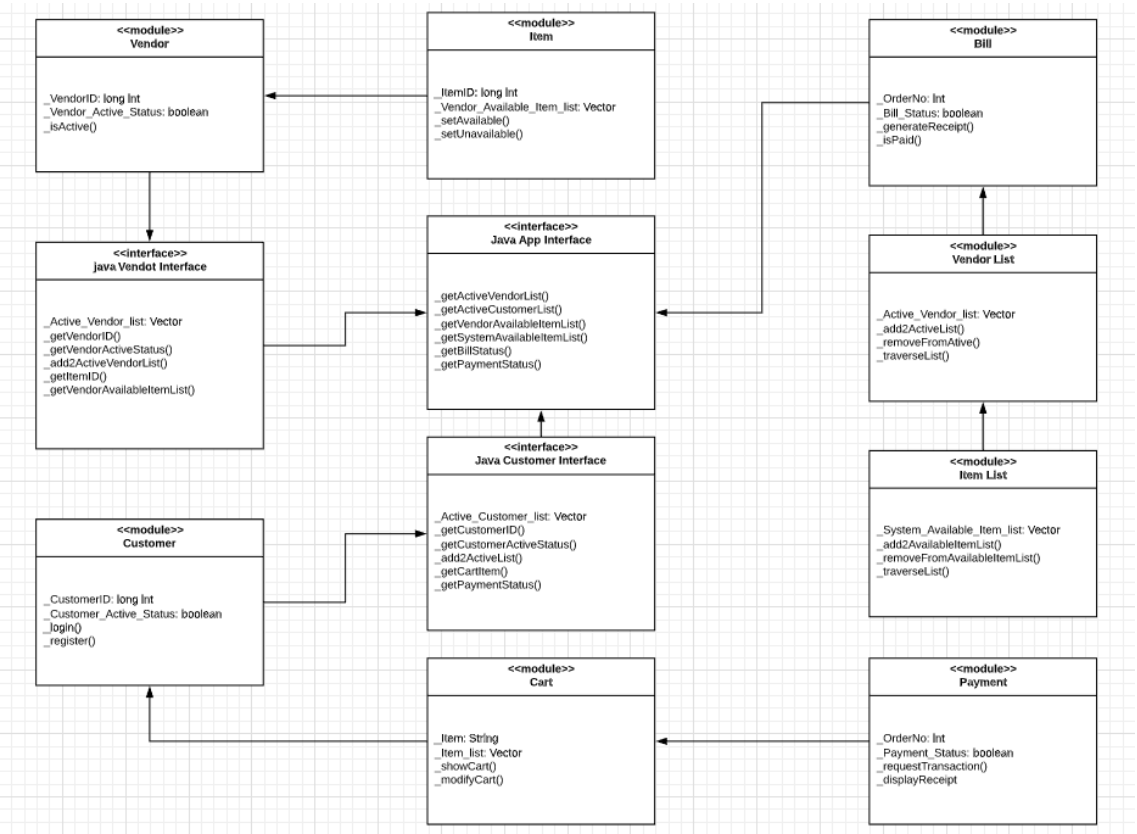
Payment:

- sendTransactionRequest(): send payment request for customer
- createReceipt(): create the receipt of the order
- displayReceipt(): display the receipt of the order

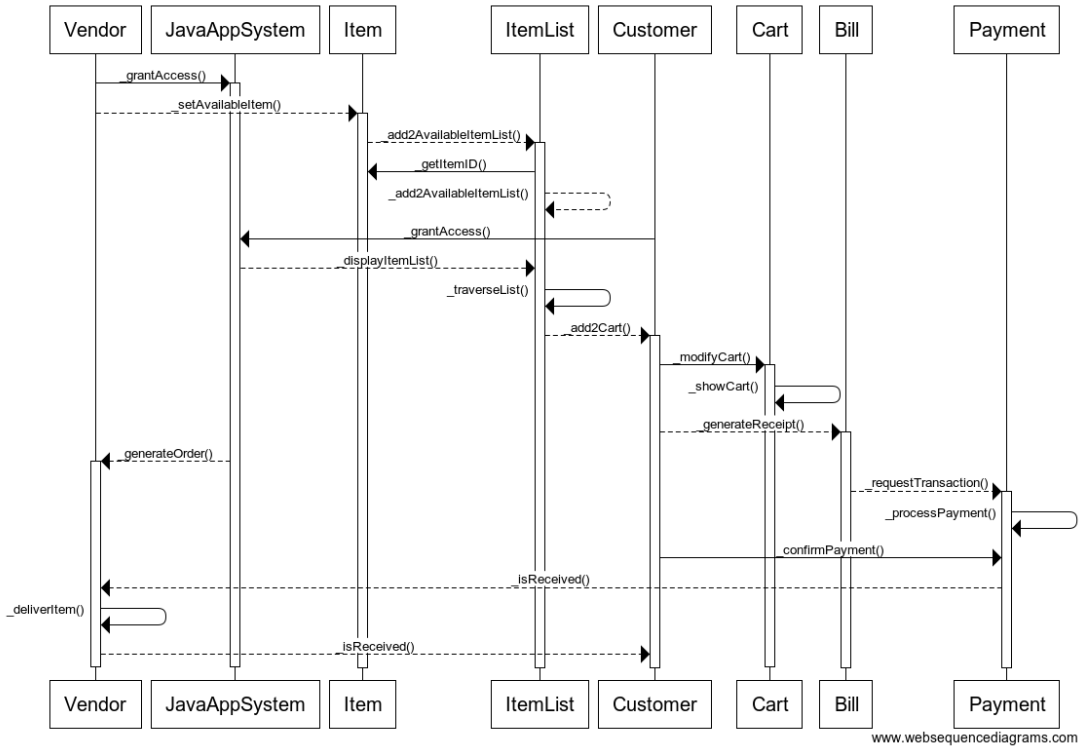
Cart:

- showCart(): display the cart
- modifyCart(): change the cart

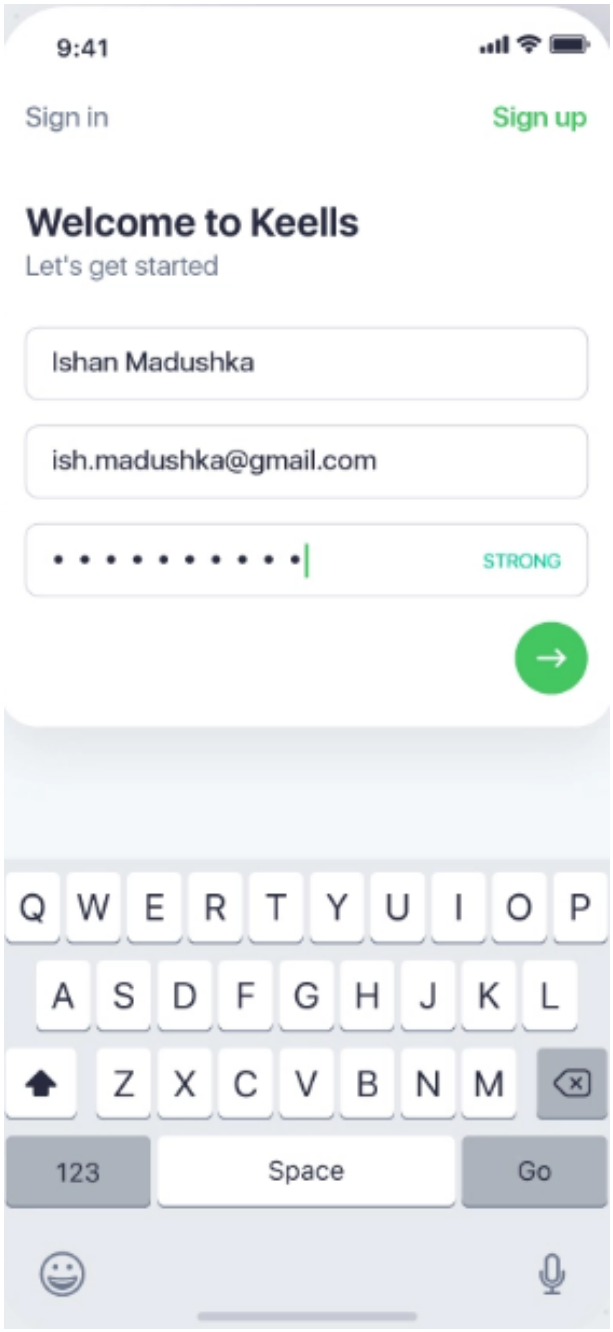
14 Module Interface

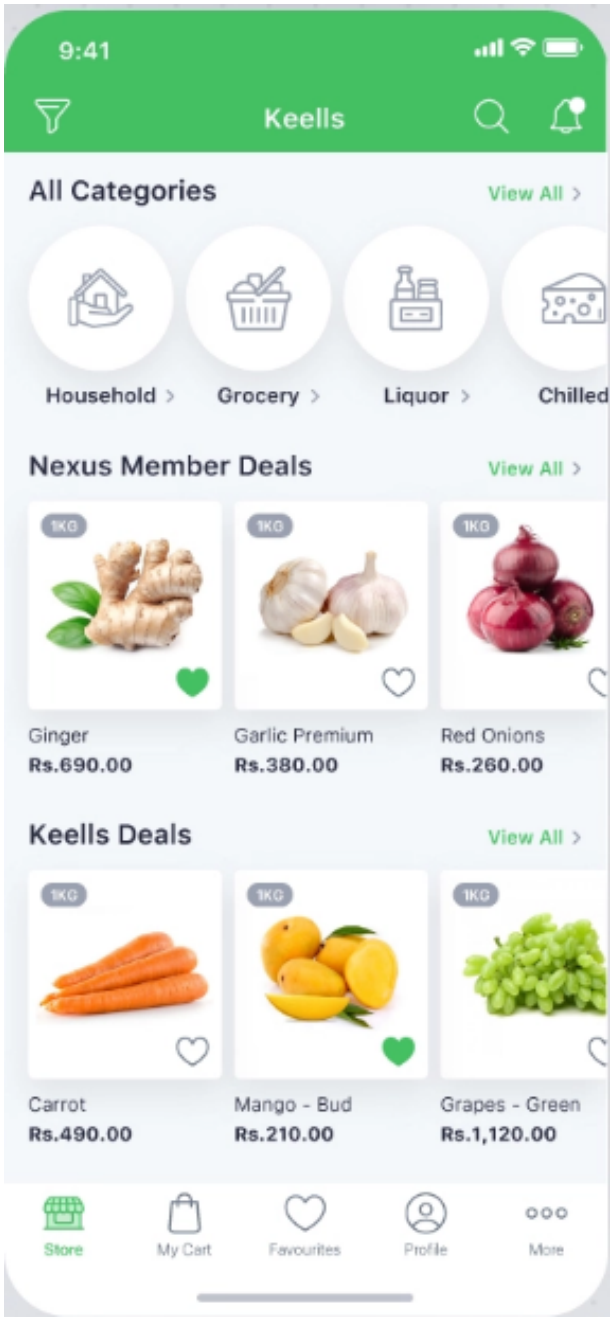


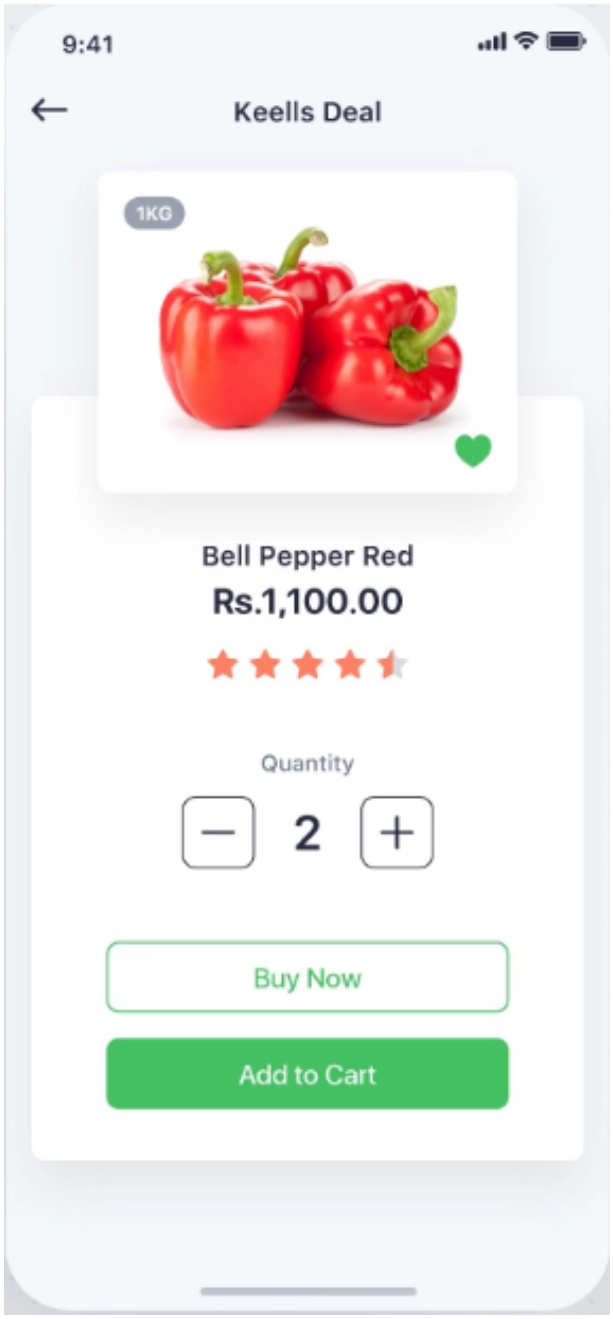
15 Sequence Diagram



16 Demonstration




9:41
Keells
All Categories
View All
Household
Grocery
Liquor
Chilled
Nexus Member Deals
View All
1KG
Ginger
Rs.690.00
Garlic Premium
Rs.380.00
Red Onions
Rs.260.00
Keells Deals
View All
Carrot
Rs.490.00
Mango - Bud
Rs.210.00
Grapes - Green
Rs.1,120.00
Store
My Cart
Favourites
Profile
More


9:41
Keells Deal
1KG
Bell Pepper Red
Rs.1,100.00
Quantity
2
Buy Now
Add to Cart