FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY

Bachelor of Software Engineering (Honours)

Programme: RSW (Group: __6__)

Assignment

BACS2083 FORMAL METHODS FOR SOFTWARE ENGINEERING

Name (Block Letters)	Photo	Registration No.	Signature	Date
1. Hing Zi Hui		22WMR05651	hui	20/9/2023
2. Ho Wen Ting		22WMR05654	ting	20/9/2023
3. Joshua Chong Zhiguang		22WMR05661	zhi	20/9/2023

4. Lee Chen Hong		22WMR05669	lee	20/9/2023
5. Lee Wee Harn		22WMR05673	Harn	20/9/2023

Tutor's Name: ___MAZLINDA BINTI NEZAM MUDEEN_____

FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY

Plagiarism Statement and Guideline for Late Submission of Coursework

Read, complete, and sign this statement to be submitted with the written report.

We confirm that the submitted work are all our own work and are in our own words.

Name (Block Letters)	Registration No.	Signature	Date
1. Hing Zi Hui	22WMR05651	hui	20/9/2023
2. Ho Wen Ting	22WMR05654	ting	20/9/2023
3. Joshua Chong Zhiguang	22WMR05661	zhi	20/9/2023
4. Lee Chen Hong	22WMR05669	lee	20/9/2023
5. Lee Wee Harn	22WMR05673	Harn	20/9/2023

BACS2083 Formal Methods for Software Engineering - Individual Presentation Rubric (202305)

Programme: RSW Group: 6	Student Names:	(1) Ho Wen Ting	(2) Hing Zi Hui	(3) Joshua Chong Zhiguang	(4) Lee Wee Harn	(5) Lee Chen Hong	
CLO3 - Propose solution using formal specifications based on requirements (A3, PLO9). (Personal Skills)							
Section	**Criteria/Area**	**Very Poor**	Rating: 1-Very Poor, 2-Poor, 3-Average, 4-Good, 5-Excellent	Rating: 1-Very Poor, 2-Poor, 3-Average, 4-Good, 5-Excellent	Rating: 1-Very Poor, 2-Poor, 3-Average, 4-Good, 5-Excellent	Rating: 1-Very Poor, 2-Poor, 3-Average, 4-Good, 5-Excellent	Rating: 1-Very Poor, 2-Poor, 3-Average, 4-Good, 5-Excellent	**Excellent**
Content: Demonstrate understanding on design	Ability to explain the ideas and design concepts correctly and provide alternative solutions.	Delivery of ideas and design concepts is vague, unorganised and without confidence.	1 2 3 4 5	1 2 3 4 5	1 2 3 4 5	1 2 3 4 5	1 2 3 4 5	Able to present ideas and design concepts very clearly, organized, attractively and confidently.
Participation: Cooperation from all members	Demonstrate excellent cooperation from team members. Ability to answer the question and provides justification.	Unable to cooperate in a group and unable to answer the question.	1 2 3 4 5	1 2 3 4 5	1 2 3 4 5	1 2 3 4 5	1 2 3 4 5	Able to cooperate with other team members and answer all questions & provides excellent understanding & justification.
Presentation: Organisation of group presentation	Good presentation skill - Natural, confident delivery that does not just convey the message but enhances it; excellent use of volume, pace etc.	Inaudible,no eye contact, mumbles the words, speaker seemed uninterested and used monotone	1 2 3 4 5	1 2 3 4 5	1 2 3 4 5	1 2 3 4 5	1 2 3 4 5	Poised, clear articulation; proper volume; steady rate; good posture and eye contact; enthusiasm;
	Materials such as words and visual elements (such as presentation slids) are highly organised with required information.	Unable to provide materials such as words and visual elements (such as presentation slides) and no visual aids used	1 2 3 4 5	1 2 3 4 5	1 2 3 4 5	1 2 3 4 5	1 2 3 4 5	Able to use materials such as visual elements or tools appropriately and attractively.
TOTAL INDIVIDUAL MARK (20 marks)			**/20**	**/20**	**/20**	**/20**	**/20**	
Comments:								

BACS2083 Formal Methods for Software Engineering - Report Assignment Rubric (202305)

Title: Shopee Application

Student Names:			**Module**		
	(1)	Ho Wen Ting	ShopeeFood	Tutorial Group:	6
	(2)	Hing Zi Hui	User	Programme:	RSW
	(3)	Joshua Chong Zhiguang	Cart		
	(4)	Lee Wee Harn	Payment		
	(5)	Lee Chen Hong	Product Management		

CLO3 - Propose solution using formal specifications based on requirements (A3, PLO9). (Personal Skills)

Section	Criteria / Area		Excellent		Good		Average		Poor		Very Poor		Score (1)	Score (2)	Score (3)	Score (4)	Score (5)
New ideas: To express ideas as a result of self-exploration *	Introduction	5	Shows attempt to solve problems with excellent new ideas	(5m)	Shows attempt to solve problems with good new ideas	(4m)	Shows attempt to solve problems with moderate new ideas	(3m)	Inadequate attempt to solve problem with weak new ideas	(2m)	Very little attempt to solve problems with weak new ideas or no new idea	(0-1m)					
	Requirements list and operations	5		(5m)		(4m)		(3m)		(2m)		(0-1m)					
	Data Abstraction (Constants, Data Types, Given Sets)	5		(5m)		(4m)		(3m)		(2m)		(0-1m)					
Effort: To show effort to investigate or search for information **	State space and Initialisation	5	Shows excellent effort to complete tasks.	(5m)	Shows good effort to complete tasks.	(4m)	Shows sufficient effort to complete tasks.	(3m)	Shows inadequate effort to complete tasks.	(2m)	Shows minimal or no effort to complete tasks.	(0-1m)					
	Operation Schema	20		(17-20m)		(13-16m)		(9-12m)		(5-8m)		(0-4m)					
	Error Scenarios	15		(13-15m)		(10-12m)		(7-9m)		(4-6m)		(0-3m)					
	Complete Schema	5		(5m)		(4m)		(3m)		(2m)		(0-1m)					
Self-learning: Self-directed learning that involves leaners relate to new information, concepts, process which are more organised and complete **	Conclusion (Self-reflection)	5	Shows excellent ability to self learn.	(5m)	Shows good ability to self learn.	(4m)	Shows sufficient ability to self learn.	(3m)	Shows inadequate ability to self learn.	(2m)	Shows minimal or no effort to self learn.	(0-1m)					
Relevance: Appropriateness and relevance of references to a task *	References and Appendices	5	Shows excellent appropriateness and relevance of reference and appendices to the task.	(5m)	Shows good appropriateness and relevance of reference and appendices to the task.	(4m)	Shows sufficient appropriateness and relevance of reference and appendices to the task.	(3m)	Shows inadequate appropriateness and relevance of reference and appendices to the task.	(2m)	Shows limited appropriateness and relevance of reference and appendices to the task.	(0-1m)					
Articulation: Able to express ideas clearly and efffectively in writing and easily understood by the reader *	Final Documentation (Report)	10	Shows excellent ability to express ideas clearly and efffectively in writing and easily understood by the reader.	(9-10m)	Shows good ability to express ideas clearly and efffectively in writing and easily understood by the reader.	(7-8m)	Shows sufficient ability to express ideas clearly and efffectively in writing and easily understood by the reader.	(5-6m)	Shows inadequate ability to express ideas clearly and efffectively in writing and easily understood by the reader.	(3-4m)	Shows limited ability to express ideas clearly and efffectively in writing and easily understood by the reader.	(0-2m)					

TOTAL INDIVIDUAL MARK (80 marks)						
Comments:						

* Group Assessment

** Individual Assessment

1.0 Background Introduction

1.1 Introduction

Shopee is one of Malaysia's most popular online shopping platforms, and as part of the project's implementation for formal methods. This project will go through a process of declaring a solution using formal specifications for the requirements of the Shopee platform. Besides, the entity relationship diagram represents how the entities around the modules play a crucial role for maintaining the functionality of the whole system.

The product management module is responsible for inventory management and CRUD (Create, Retrive, Update, Delete) operations for products. The new product will be tested by verifying its existence in this module.

The user module is essential for allowing users to use the application's features. The user module mainly is responsible for users to perform their registration , view and update their profile. Moreover, the login phase is crucial for users to key in their personal information such as password to successfully login and make any purchase inside the Shopee Application. Furthermore, there is also a purchase history function available for the user to view the product that he/she purchased so that the buyer is able to trace their product information.

The cart module is responsible for managing items added by the user and performing a search function. The testing team will verify the validity of these actions and check that the item remains in the cart.

The Shopee provides various payment methods, payment verification, and gateway error handling to ensure the user's security at all times. Additionally, the application offers a SpayLater method, enabling users to pay in installments within a specified limit.

Finally, the Shopee Food Ordering module within the application allows users to add their desired food from the restaurant into their food basket. The item inside the food basket is

allowed to be read, updated or removed by the user. Users also are able to add their desired restaurant into the favorite list which also can be removed from the list. All the orders that consisted of different states will also be able to retrieve from the system.

1.2 Entity Relationship Diagram (ERD)

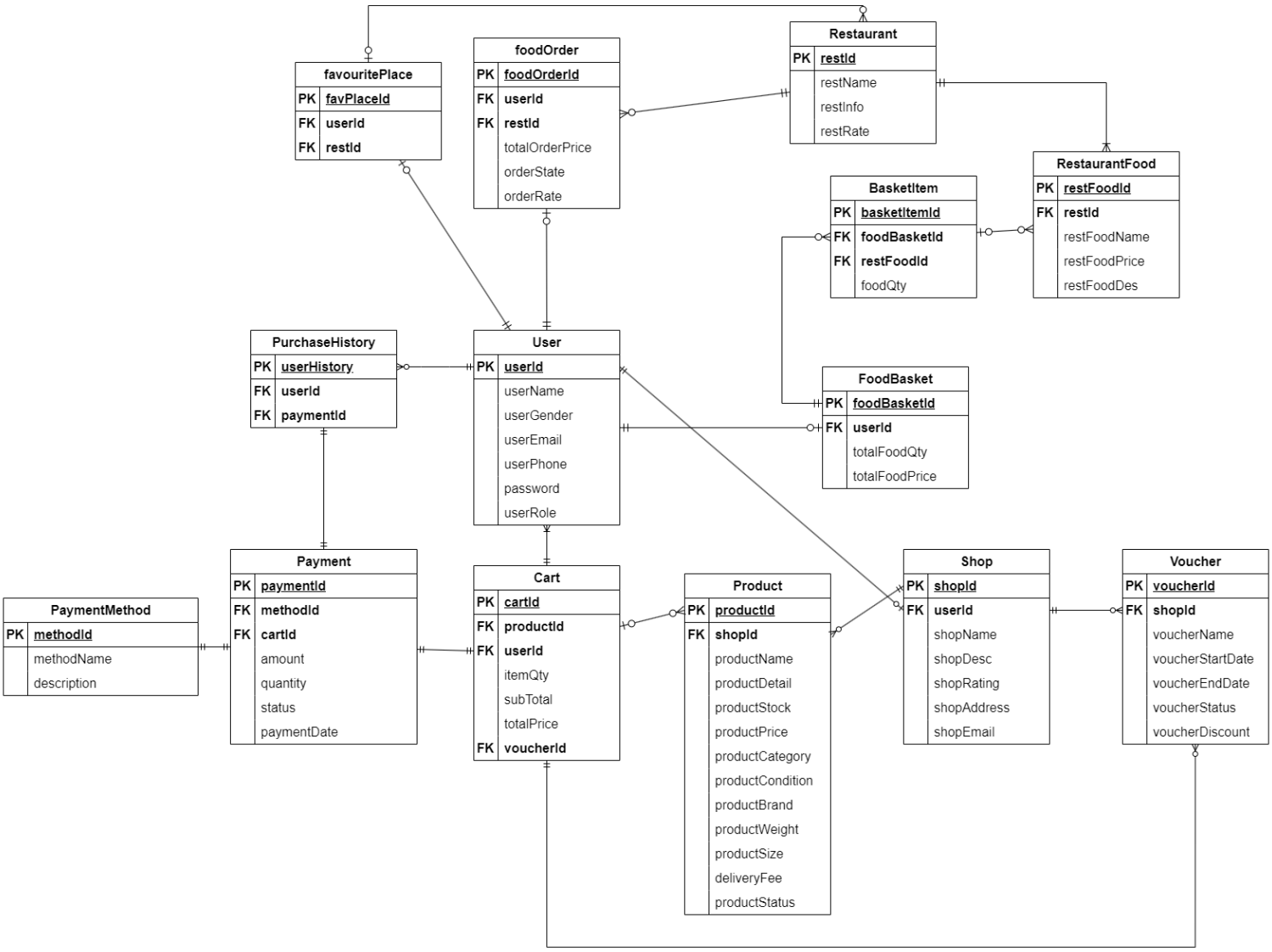

Diagram 1.2.1 ERD for Shopee

2.0 Requirements List and Operations

2.1 User Module: Hing Zi Hui

Operation 1: Create, Update & Retrieve Buyer

- The system shall allow users to register themself as buyers.
- The system shall allow users to view their user profile.
- The system shall allow users to update their user information.

Operation 2: Login & Logout Buyer

- The system shall allow users to login and make purchases.
- The system shall allow users to logout from the system

Operation 3: View Buy History

- The system shall allow users to view their purchase history through the system.

2.2 Product Management Module: Lee Chen Hong

Operation 1: Create, Update, Retrieve & Delete Product

- The system shall allow users to create products.
- The system shall allow users to view products information
- the system shall allow users to update products information.
- The system shall allow users to delete products.

Operation 2: Create, Update, Retrieve & Delete Voucher

- The system shall allow users to create vouchers.
- The system shall allow users to view voucher information
- the system shall allow users to update voucher information.
- The system shall allow users to delete vouchers.

Operation 3: Create, Update, Retrieve & Delete Shop

- The system shall allow users to create vouchers.
- The system shall allow users to view voucher information
- the system shall allow users to update voucher information.
- The system shall allow users to delete vouchers.

2.3 Cart Module: Joshua Chong ZhiGuang

Operation 1: Create, Retrieve & Delete Cart

- The system shall allow the user to view the added item inside the cart
- The system shall allow the user to remove the item from the cart.
- The system shall allow the user to edit the added item quantity.

Operation 2: Apply Voucher

- The system shall allow the user to apply the voucher.-

Operation 3: Redeem Shopee Coin

- The system shall allow the user to redeem the shopee coin.

2.4 Payment Module : Lee Wee Harn

Operation 1: Make Payment

- The system shall display the total amount that users need to pay.
- The system shall allow users to enter their delivery address.
- The system shall allow users to change their delivery address.
- The system shall allow users to select payment methods.

Operation 2: View Payment

- The system shall display the payment status indicating whether the payment is pending, completed, declined, or refunded.
- The system shall allow users to view payment history.

Operation 3: Add Payment Method

- The system shall support various payment methods such as credit/debit cards, online banking and E-wallets.
- The system shall allow users to enter the relevant information for the chosen payment method, such as card details, bank account numbers, or E-wallet credentials.

2.5 ShopeeFood Module: Ho Wen Ting

Operation 1: Create, Retrieve, Update & Delete Food Basket

- The system shall allow users to add products into the food basket.
- The system shall allow users to view all the existing products inside the food basket.
- The system shall allow users to update the product's details in the food basket.
- The system shall allow users to remove the products from the food basket.

Operation 2: Create, Retrieve & Delete Favorite Places

- The system shall allow users to add their favorite places into the favorite list.
- The system shall allow users to view all their favorite places inside the favorite list.
- The system shall allow users to remove their favorite places from the favorites list.

Operation 3: View Order

- The system shall allow users to view all their orders.
- The system shall allow users to view the specific orders based on the state of the order.
- The system shall allow users to rate the completed order.

3.0 Data Abstraction

3.1 Basic Type

User

[USERID] - The set of all user's ID.

[USERNAME] - The set of all user's full names.

[USEREMAIL] - The set of all user's email addresses.

[USERPHONE] - The set of all user's phone numbers.

[PASSWORD] - The set of all user's passwords.

[USERHISTORY] - The set of all user's purchase history.

Shop

[SHOPID] - The set of all stops's id.

[SHOPNAME] - The set of all shops' names.

[SHOPDESC] - The set of all shops' descriptions.

[SHOPADDRESS] - The set of all shops' addresses.

SHOPRATING = $\mathbb{N}$

Voucher

[VOUCHERID] - The set of all vouchers' id

[VOUCHERNAME] - The set of all vouchers' names.

[VOUCHERSTARTDATE] - The set of all vouchers' start dates.

[VOUCHERENDDATE] - The set of all vouchers' end dates.

VOUCHERDISCOUNT = $\mathbb{N}$

Product

[PRODUCTID] - The set of all product IDs.

[PRODUCTNAME] - The set of all product names.

[PRODUCTDETAIL]- The set of all product details.

[PRODUCTCATEGORY] – The set of all product category.

[PRODUCTBRAND] – The set of all product brand

PRODUCTSTOCK = ℕ

PRODUCTPRICE = ℕ

PRODUCTWEIGHT = ℕ

PRODUCTSIZE = ℕ

DELIVERYFEE = ℕ

Cart

[CARTID] - The set of all cart IDs.

SHOPEECOIN = ℕ

ITEMQTY = ℕ

TOTALPRICE = ℕ

SUBTOTAL = ℕ

Payment

[PAYMENTID] - The set of all payment's ID.

[PAYMENTDATE] - The set of all payment dates.

[DELIVERYADDRESS] - The set of all delivery addresses.

[PAYMENTAMOUNT] - ℕ

Payment Method

[METHODID] - The set of all payment method's ID.

[METHODNAME] - The set of all payment method's names.

[MOTHODDESCRIPTION] - The set of all payment method's description.

Restaurant

[RESTID] - The set of all restaurant's IDs.

[RESTNAME] - The set of all restaurant's name.

[RESTINFO] - The set of all restaurant's information.

RESTRATE = ℕ

Restaurant Food

[RESTFOODID] - The set of all restaurant's products' IDs.

[RESTFOODNAME] - The set of all restaurant's products' names.

[RESTFOODDES] - The set of all restaurant's products' descriptions.

RESTFOODPRICE = $\mathbb{N}$

Food Basket

[FOODBASKETID] - The set of all user's food basket IDs.

TOTALFOODQTY = $\mathbb{N}$

TOTALFOODPRICE = $\mathbb{N}$

Basket Item

[BASKETITEMID] - The set of all user's food basket items' IDs.

FOODQTY = $\mathbb{N}$

Food Order

[FOODORDERID] - The set of all user's food order IDs.

[TOTALORDERPRICE] = $\mathbb{N}$

[ORDERRATE] = $\mathbb{N}$

Favorite Place

[FAVPLACEID] - The set of all user's favorite place IDs.

3.2 Free Type

Voucher

VOUCHERSTATUS ::= used | invalid | valid

Product

PRODUCTCONDITION ::= new | used

PRODUCTSTAATUS ::= invalid | live | soldOut | delist

VOUCHERSTATUS ::= used | invalid | valid

User

USERGENDER ::= male | female

USERROLE ::= buyer | seller

USERSTATUS ::= loggedOut | loggedIn

Payment

PAYMENTMETHOD ::= credit card | debit card | online banking | E-wallet

PAYMENTSTATUS ::= successful | unsuccessful

Food Order

ORDERSTATE ::= processing | complete | refund | to rate

3.3 Axiomatic Definition

Product

$$maxQuantity : \mathbb{N}$$

$$maxQuantity = 9999$$

$$maxRating : \mathbb{N}$$

$$maxRating = 5$$

Cart

$$maxAddCart = \mathbb{N}$$

$$maxAddCart < 20$$

Payment

$$maxPaymentAmount = \mathbb{N}$$

$$maxPaymentAmount \leqslant paymentAmount$$

Basket Item

$$minFoodQty = \mathbb{N}$$

$$minFoodQty = 1$$

Favorite Place

$$maxFavorite = \mathbb{N}$$

$$maxFavorite = 100$$

Restaurant

minRate = *N*
minRate = 0

maxRate = ℕ
maxRate = 5

4.0 User Module

4.1 State Space Schema

User

$userId : \mathbb{P}\ USERID$

$userName : \mathbb{P}\ (USERID \times USERNAME)$

$userEmail : \mathbb{P}\ (USERID \times USEREMAIL)$

$userGender: \mathbb{P}\ (USERID \times USERGENDER)$

$userPhone : \mathbb{P}\ (USERID \times USERPHONE)$

$userPassword : \mathbb{P}\ (USERID \times PASSWORD)$

$userHistory : \mathbb{P}\ (USERID \times USERHISTORY)$

$userStatus: \mathbb{P}\ (USERID \times USERSTATUS)$

---

dom userName = dom userEmail = dom userGender = dom userPhone
= dom password = dom userHistory = dom userStatus = userId

4.2 Initial State Schema

InitUser

User
userId = ∅
userName = ∅
userEmail = ∅
userGender = *male*
userPhone = ∅
password = ∅
userHistory = ∅
userStatus = *loggedOut*

4.3 Operation Schema

Operation 1: Create, Update & Retrieve Buyer

UserRegistration

Δ *User*

user? : *USERID*
name? : *USERNAME*
email? : *USEREMAIL*
gender? : *USERGENDER*
phone? : *USERPHONE*
password? : *PASSWORD*

(user? ∉ userId ∧
name? ∉ ran *userName* ∧
email? ∉ ran *userEmail* ∧
phone? ∉ ran *userPhone)*

userId′ = userId ∪ *{user?}*
userName′ = userName ∪ *{user? ↦ name?}*
userEmail′ = userEmail ∪ *{user? ↦ email?}*
userGender′ = userGender ∪ *{user? ↦ gender?}*
userPhone′ = userPhone ∪ *{user? ↦ phone?}*
userPassword′ = userPassword ∪ *{user? ↦ password?}*

ViewProfile

Ξ *User*

user? : USERID
name! :USERNAME
email! : USEREMAIL
gender! : USERGENDER
phone! : USERPHONE

(user? ∈ userId ∧ userStatus (|{user?}|) = loggedIn)

user! = userId
name! = userName (|{user?}|)
email! = userEmail (|{user?}|)
gender! = userGender (|{user?}|)
phone! = userPhone (|{user?}|)

UpdateProfile

Δ *User*

user? : USERID
name? :USERNAME
email? : USEREMAIL
gender? : USERGENDER
phone? : USERPHONE
password? : PASSWORD

(user? ∈ userId ∧ userStatus (|{user?}|) = loggedIn)

name? ∉ ran *userName* ∨
email? ∉ ran *userEmail* ∨
phone? ∉ ran *userPhone*

userName′ = userName ⊕ {user? ↦ name?}∨
userEmail′ = userEmail ⊕ {user? ↦ email?}∨
userGender′ = userGender ⊕ {user? ↦ gender?}∨
userPhone′ = userPhone ⊕ {user? ↦ phone?}∨
userPassword′ = userPassword ⊕ {user? ↦ password?}

Operation 2: Login & Logout Buyer

LogIn

Δ *User*

name? : USERNAME
email? : USEREMAIL
password? : PASSWORD

(name? ∈ ran *username ∨ email? ∈* ran *userEmail)*
∧
(password? ∈ userPassword (|{dom (userName ▷ {name?})}|)
∨
password? ∈ userPassword (|{dom (userEmail▷ {email?})}|))
∧
(userStatus(|{dom (userName ▷ {name?})}|) = loggedOut
∨
userStatus(|{dom (userEmail ▷ {email?})}|) = loggedOut
)

userStatus' = userStatus ⊕ {dom (userName ▷ {name?}) ↦ loggedIn} ∨
userStatus' = userStatus ⊕ {dom (userEmail ▷ {email?}) ↦ loggedIn}

Logout

ΔUser
user? : USERID

---

userStatus (|{user?}|) = loggedIn

userStatus' = userStatus ⊕ {user? ↦ loggedOut}

Operation 3: View Buy History

ViewBuyHistory

Ξ *User*
Ξ *Payment*
Ξ *PaymentWithUser*
Ξ *PaymentWithCart*
Ξ *Cart*
Ξ *Product*

user? : *USERID*
paid? : PAYMENTID
cartID! : CARTID
productID! : PRODUCTID
prodName! : PRODUCTNAME
prodDetail! : PRODUCTDETAIL
prodCategory! : PRODUCTCATEGORY
status! : PAYMENTSTATUS
amount! : PAYMENTAMOUNT
date! : PAYMENTDATE

(user? ∈ userId ∧ userStatus(|{user?}|) = loggedIn)
(paid? ∈ paymentUserId~(|{user?}|) ∧ paid? ∈ dom *paymentCartId)*

cartID! = *paymentCartId*(|{*paid?*}|)
productID! = *cartProductId* ▷ { *paymentCartId* (|{*paid?*}|) }
prodName! = prodName (|{ *cartProductId* ▷ { *paymentCartId* (|{*paid?*}|) }|)
prodDetail! = prodDetail (|{ *cartProductId* ▷ { *paymentCartId* (|{*paid?*}|) }|)
prodCategory! = prodCategory (|{ *cartProductId* ▷ { *paymentCartId* (|{*paid?*}|) }|)
status! = paymentStatus(paid?)
amount! = paymentAmount(paid?)
date! = paymentDate(paid?)

4.4 Error Scenario

Error Scenario Table

Operation 1: Create, Retrieve, Update & Delete Food Basket

Schema Nme	Success Pre-Condition	Failure Pre-Condition	Remark
UserRegistration	*user? ∉ userId* *name? ∉ ran userName* *email? ∉ ran userEmail* *phone? ∉ ran userPhone*	*user? ∈ userId* *name? ∈ ran userName* *email? ∈ ran userEmail* *phone? ∈ ran userPhone*	UserAlreadyExist
ViewProfile	*user? ∈ userId* *userStatus(|{user?}|)* *= loggedIn*	userId? ∉ userId userStatus ≠ loggedIn	UserNotExist UserNotLogin
UpdateProfile	*user? ∈ userId* *userStatus(|{user?}|)* *= loggedIn* *name? ∉ ran userName* *email? ∉ ran userEmail* *phone? ∉ ran userPhone*	*user? ∉ userId* *userStatus(|{user?}|)* *= loggedOut* *name? ∈ ran userName* *email? ∈ ran userEmail* *phone? ∈ ran userPhone*	UserNotExist UserNotLogin UserAlreadyExist
Login	*name? ∈ ran userName* *email? ∈ ran userEmail* *password? ∈ userPassword (|{dom (userName ▷ {name?})}|)* *password? ∈ userPassword (|{dom (userName ▷ {name?})}|)* *userStatus{dom (userName ▷ {name?})} = loggedOut* *userStatus{dom (userEmail ▷ {email?})} = loggedOut*	*me? ∉ ran userName* *email? ∉ ran userEmail* *password? ∉ userPassword (|{dom (userName ▷ {name?})}|)* *password? ∉ userPassword (|{dom (userName ▷ {name?})}|)* *userStatus{dom (userName ▷ {name?})} = loggedIn* *userStatus{dom (userEmail ▷ {email?})} = loggedIn*	UserNotExist UserAlreadyLogin InvalidPassword

LogOut	$userStatus\ (\!|\{user?\}|\!)$ $= loggedIn$	$userStatus\ (\!|\{user?\}|\!)$ $= loggedOut$	UserNotLogin
ViewBuyHistory	$user? \in userId$ $userStatus\ (\!|\{user?\}|\!)$ $= loggedIn$ $paid? \in UserID(\!|\{user?\}|\!)$	$user? \notin userId$ $userStatus\ (\!|\{user?\}|\!)$ $= loggedOut$ $paid? \notin UserID(\!|\{user?\}|\!)$	UserNotExist UserNotLogin PaymentNotExist

Error Handling Free Type:

REPORT ::=OKMessage | *UserAlreadyExist* | *UserNotExist* | *UserNotLogin* | *UserAlreadyLogin*

| *InvalidPassword* | *PaymentNotExist*

OKMessage

report! : *REPORT*

---

report! = *OKMesssage*

UserAlreadyExist

Ξ*User*
user? : *USERID*
email?: USEREMAIL
phone?: USERPHONE
report! : *REPORT*

---

user? ∈ *userId* ∨
name? ∈ ran *userName* ∨
email? ∈ ran *userEmail* ∨
phone? ∈ ran *userPhone*

report! = *UserAlreadyExist*

UserNotExist

Ξ*User*
user? : *USERID*
name? : USERNAME
email?: USEREMAIL
phone?: USERPHONE
report! : *REPORT*

---

user? ∉ *userId* ∨
name? ∉ ran *userName* ∨
email? ∉ ran *userEmail* ∨
phone? ∉ ran *userPhone*
report! = *UserNotExist*

UserNotLogin

$\Xi User$

user? : *USERID*

report! : *REPORT*

(user $\in$ *userId* $\wedge$ *userStatus* $(\!|\{user?\}|\!)$ = *loggedOut)*

report! = *UserNotLogin*

UserAlreadyLogin

$\Xi User$

user? : *USERID*

report! : *REPORT*

(user $\in$ *userId* $\wedge$ *userStatus* $(\!|\{user?\}|\!)$ = *loggedIn)*

report! = *UserAlreadyLogin*

InvalidPassword

$\Xi User$

user? : *USERID*

password? : *PASSWORD*

---

user $\in$ *userId* $\wedge$ *password?* $\notin$ *userPassword* $(\!|\{user?\}|\!)$

report! = *InvalidPassword*

PaymentNotExist

Ξ *User*

Ξ *PaymentWithUser*

user? : *USERID*

paid? : PAYMENTID

(user? $\in$ *userId* $\wedge$ *userStatus* $(\!|\{user?\}|\!)$ = *loggedIn)*

paid? $\notin$ *UserID*$(\!|\{user?\}|\!)$

report! = *PaymentNotExist*

Error Schema 1 : Create, Update & Retrieve Buyer

UserRegistrationError

Ξ *User*
user? : *USERID*
email? : *USEREMAIL*
phone? : *USERPHONE*
rep!: REPORT

(user? $\in$ *userId*
$\vee$
name? $\in$ ran *userName*
$\vee$
email? $\in$ ran *userEmail*
$\vee$
phone? $\in$ ran *userPhone)* $\wedge$ *rep! = UserAlreadyExist*

ViewProfileError

Ξ *User*
user? : *USERID*
rep!: REPORT

---

user? $\notin$ *userId* $\wedge$ *rep! = UserNotExist*
$\vee$
userStatus$(\{$*user?*$\})$ = *loggedOut* $\wedge$ *rep! = UserNotLogin*

UpdateProfileError

Ξ *User*
user? : *USERID*
name? : USERNAME
email? : *USEREMAIL*
phone? : *USERPHONE*
rep!: REPORT

---

user? $\notin$ *userId* $\wedge$ *rep! = UserNotExist*
$\vee$
((name? $\in$ ran *userName*
$\vee$
email? $\in$ ran *userEmail*
$\vee$
phone? $\in$ ran *userPhone)* $\wedge$ *rep! = UserAlreadyExist)*

Error Schema 2 : Login & Logout Buyer

LogInError

Ξ *User*
name? :USERNAME
email? : USEREMAIL
password? : PASSWORD
rep!: REPORT

((name? $\notin$ ran *username*
$\vee$
email? $\notin$ ran *userEmail)* $\wedge$ *rep! = UserNotExist)*
$\vee$
((password? $\notin$ *userPassword (|(dom (userName* $\rhd$ *{name?}))|)*
$\vee$
password? $\notin$ *userPassword (|(dom (userEmail* $\rhd$ *{email?}))|)* $\wedge$ *rep! = InvalidPassword)*
$\vee$
((*userStatus(|{dom (userName* $\rhd$ *{name?})}|) = loggedIn*
$\vee$
userStatus(|{dom (userEmail $\rhd$ *{email?})}|) = loggedIn)* $\wedge$ *rep! = UserAlreadyLogin)*

LogOutError

Ξ *User*
user? : *USERID*
rep!: REPORT

---

userStatus(|{user?)}|) = loggedOut $\wedge$ *rep! = UserNotLogin*

Error Schema 3 : View Buy History

ViewBuyHistoryError

Ξ *User*

Ξ *PaymentWithUser*

user? : *USERID*

paid? : PAYMENTID

$user \notin userId \land rep! = UserNotExist$

$\lor$

$userStatus(\{user?)\}) = loggedOut \land rep! = UserNotLogin$

$\lor$

$paid? \notin UserID(\{user?\}) \land rep! = PaymentNotExist$

4.5 Complete Schema

Complete 1: Create, Update & Retrieve Buyer

UserRegistrationComplete ≙ (UserRegistration ∧ OKMessage) ∨ UserRegistrationError

UserRegistrationComplete

userId, userId' : ℙ *USERID*
userName, userName' : ℙ (*USERID* × *USERNAME*)
userEmail, userEmail' : ℙ (*USERID* × *USEREMAIL*)
userGender, userGender': ℙ *(USERID* × *USERGENDER)*
userPhone, userPhone' : ℙ *(USERID* ×*USERPHONE)*
userPassword, userPassword', : ℙ *(USERID* ×*PASSWORD)*
userHistory, userHistory' : ℙ *(USERID* ×*USERHISTORY)*
userStatus, userStatus': ℙ *(USERID* × *USERSTATUS)*
user? : *USERID*
name? : *USERNAME*
email? : *USEREMAIL*
gender? : *USERGENDER*
phone? : *USERPHONE*
password? : *PASSWORD*
rep! : REPORT

(user? ∉ *userId* ∧ *name?* ∉ ran *userName* ∧ *email?* ∉ ran *userEmail* ∧ *phone?* ∉ ran *userPhone* ∧
userId' = *userId* ∪ {*user?*} ∧ *userName'* = *userName* ∪ {*user?* ↦ *name?*} ∧ *userEmail'* = *userEmail* ∪
{*user?* ↦ *email?*} ∧*userGender'* = *userGender* ∪ {*user?* ↦ *gender?*} ∧ *userPhone'* =
userPhone ∪ {*user?* ↦ *phone?*} ∧ *userPassword'* = *userPassword* ∪ {*user?* ↦ *password? }* ∧
rep! = *OKMesssage* ∧ dom *userName* = dom *userEmail* = dom *userGender* = dom *userPhone*
= dom *userPassword* = *userId* ∧ dom *userName'* = dom *userEmail'* = dom *userGender'* = dom *userPhone'*
= dom *userPassword'* = *userId')*
∨
(((user? ∈ *userId*
∨
name? ∈ ran *userName*
∨
email? ∈ ran *userEmail*
∨
phone? ∈ ran *userPhone) ∧ rep! = UserAlreadyExist* ∧ dom *userName* = dom *userEmail* =
dom *userGender* = dom *userPhone* = dom *userPassword* = *userId* ∧ dom *userName'* = dom *userEmail'*
= dom *userGender'*= dom *userPhone'* = dom *userPassword'*= *userId')*
∧

(userId'= userId ∧ userName'=userName ∧ userEmail' = userEmail ∧ userGender' = userGender ∧ userPhone' = userPhone ∧ userPassword'= userPassword))

ViewProfileComplete ≙ (ViewProfile ∧ OKMessage) ∨ ViewProfileError

ViewProfileComplete

$userId, userId' : \mathbb{P}\ USERID$
$userName, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userEmail, userEmail' : \mathbb{P}\ (USERID \times USEREMAIL)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone, userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword', : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$
$user? : USERID$
$name! : USERNAME$
$email! : USEREMAIL$
$gender! : USERGENDER$
$phone! : USERPHONE$
$rep! : REPORT$

$((user? \in userId \curlywedge userStatus(\!|\{user?\}|\!) = loggedIn) \land (user! = userId \land name! = userName\ (\!|\{user?\}|\!) \land$
$email! = userEmail\ (\!|\{user?\}|\!) \land gender! = userGender\ (\!|\{user?\}|\!) \land phone! = userPhone\ (\!|\{user?\}|\!)) \land$
$rep! = OKMesssage \land \text{dom}\ userName = \text{dom}\ userEmail = \text{dom}\ userGender = \text{dom}\ userPhone =$
$\text{dom}\ userPassword = \text{dom}\ userStatus = userId \land \text{dom}\ userName' = \text{dom}\ userEmail' = \text{dom}\ userGender' = \text{dom}\ us$
$= \text{dom}\ userPassword' = \text{dom}\ userStatus' = userId')$
$\lor$
$(((user? \notin userId \land rep! = UserNotExist \land userId \land userId')$
$\lor$
$(userStatus(\!|\{user?\}|\!) = loggedOut \land rep! = UserNotLogin \land \text{dom}\ userStatus = userId \land$
$\text{dom}\ userStatus' = userId'))$
$\land$
$(userId' = userId \land userStatus' = userStatus))$

UpdateProfileComplete ≙ (UpdateProfile∧ OKMessage) ∨ UpdateProfileError

UpdateProfileComplete

$userId, userId' : \mathbb{P}\ USERID$
$userName, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userEmail, userEmail' : \mathbb{P}\ (USERID \times USEREMAIL)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone, userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword', : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$
$user? : USERID$
$name? : USERNAME$
$email? : USEREMAIL$
$gender? : USERGENDER$
$phone? : USERPHONE$
$password? : PASSWORD$
$rep! : REPORT$

$((user? \in userId \land userStatus(\!|\{user?\}|\!) = loggedIn) \land (name? \notin \text{ran } userName \lor email? \notin \text{ran } userEmail \lor phone? \notin \text{ran } userPhone) \land (userName' = userName \oplus \{user? \mapsto name?\} \lor userEmail' = userEmail \oplus \{user? \mapsto email?\} \lor userGender' = userGender \oplus \{user? \mapsto gender?\} \lor userPhone' = userPhone \oplus \{user? \mapsto phone?\} \lor userPassword' = userPassword \oplus \{user? \mapsto password?\}) \land rep! = OKMesssage \land$
$\text{dom } userName = \text{dom } userEmail = \text{dom } userGender = \text{dom } userPhone = \text{dom } userPassword = \text{dom } userStatus = userId \land \text{dom } userName' = \text{dom } userEmail' = \text{dom } userGender' = \text{dom } userPhone' = \text{dom } userPassword' = \text{dom } userStatus' = userId')$
$\lor$
$(((user? \notin userId \land rep! = UserNotExist \land userId \land userId')$
$\lor$
$(userStatus(\!|\{user?\}|\!) = loggedOut \land rep! = UserNotLogin \land \text{dom } userStatus = userId \land \text{dom } userStatus' = userId')$
$\lor$
$((name? \in \text{ran } userName$
$\lor$
$email? \in \text{ran } userEmail$
$\lor$
$phone? \in \text{ran } userPhone) \land rep! = UserAlreadyExist \land \text{dom } userName = \text{dom } userEmail = \text{dom } userGender = \text{dom } userPhone \text{ dom } userStatus = userId \land \text{dom } userStatus' = userId'))$
$\land$
$(userId' = userId \land userStatus' = userStatus \land userName' = userName \land userEmail' = userEmail \land userPhone' = userPhone))$

Complete 2: Login & Logout Buyer

LogInComplete ≙ (LogIn ∧ OKMessage) ∨ LogInError

LogInComplete

userId, userId' : ℙ *USERID*
userName, userName' : ℙ (*USERID* × *USERNAME*)
userEmail, userEmail' : ℙ (*USERID* × *USEREMAIL*)
userGender, userGender': ℙ *(USERID* × *USERGENDER)*
userPhone, userPhone' : ℙ *(USERID* ×*USERPHONE)*
userPassword, userPassword', : ℙ *(USERID* ×*PASSWORD)*
userHistory, userHistory' : ℙ *(USERID* ×*USERHISTORY)*
userStatus, userStatus': ℙ *(USERID* × *USERSTATUS)*

name? :*USERNAME*
email? : *USEREMAIL*
password? : *PASSWORD*
rep! : *REPORT*

((name? ∈ ran *username* ∨ *email?* ∈ ran *userEmail)* ∧ *(password?* ∈ *userPassword*
(|dom (userName ▷ *{name?})|)* ∨ *password?* ∈ *userPassword (|dom (userEmail* ▷ *{email?})|))*
∧ *(userStatus(|dom (userName* ▷ *{name?})|) = *loggedOut* ∨ *userStatus(|dom (userEmail* ▷ *{email?})|)*
= *loggedOut*) ∧ (*userStatus'* = *userStatus* ⊕ *{dom (userName* ▷ *{name?})* ↦ *loggedIn}* ∨
userStatus' = *userStatus* ⊕ *{dom (userEmail* ▷ *{email?})* ↦ *loggedIn})* ∧ *rep! = OKMesssage* ∧
dom *userName* = dom *userEmail* = dom *userPassword* = *userId* ∧ dom *userName'* = dom *userEmail'* =
= dom *userPassword'* = *userId')*
∨
((((name? ∉ ran *username*
∨
email? ∉ ran *userEmail)* ∧ *rep! = UserNotExist* ∧ dom *userName* = dom *userEmail* = *userId* ∧
dom *userName'* = dom *userEmail'* = *userId')*
∨
((password? ∉ *userPassword (|dom (userName* ▷ *{name?}))|)*
∨
password? ∉ *userPassword (|dom (userEmail* ▷ *{email?}))|)*∧ *rep! = InvalidPassword* ∧ dom *userName*
= dom *userEmail* = dom *userPassword* = *userId* ∧ dom *userName'* = dom *userEmail'* = dom *userPassword'*
= *userId')*
∨
((userStatus(|dom (userName ▷ *{name?})|)* = *loggedIn*
∨

$userStatus(\!|\{dom\ (userEmail \rhd \{email?\})\}|\!) = loggedIn) \land rep! = UserAlreadyLogin \land \text{dom}\ userName = \text{dom}\ userEmail = \text{dom}\ userStatus = userId \land \text{dom}\ userName' = \text{dom}\ userEmail' = \text{dom}\ userStatus' = userId')$

$\land$

$(userId' = userId \land userStatus' = userStatus \land userName' = userName \land userEmail' = userEmail \land userPassword' = userPassword))$

LogOutComplete $\triangleq$ (LogOut $\land$ OKMessage) $\lor$ LogOutError

LogOutComplete

$userId, userId' : \mathbb{P}\ USERID$

$userName, userName' : \mathbb{P}\ (USERID \times USERNAME)$

$userEmail, userEmail' : \mathbb{P}\ (USERID \times USEREMAIL)$

$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$

$userPhone, userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$

$userPassword, userPassword', : \mathbb{P}\ (USERID \times PASSWORD)$

$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$

$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$user? : USERID$

$rep! : REPORT$

$(userStatus\ (\!|\{user?\}|\!) = loggedIn \land userStatus' = userStatus \oplus \{user? \mapsto loggedOut\} \land rep! = OKMesssage \land \text{dom}\ userStatus = userId \land \text{dom}\ userStatus' = userId')$

$\lor$

$((userStatus(\!|\{user?)\}|\!) = loggedOut \land rep! = UserNotLogin \land \text{dom}\ userStatus = userId \land \text{dom}\ userStatus' = userId')$

$\land$

$(userId' = userId \land userStatus' = userStatus))$

Complete 3: View Buy History

ViewBugHistoryError ≙ (ViewBuyHistory ∧ OKMessage) ∨ ViewBuyHistoryError

_ViewBuyHistoryComplete

userId, userId' : ℙ *USERID*
userName, userName' : ℙ (*USERID* × *USERNAME*)
userEmail, userEmail' : ℙ (*USERID* × *USEREMAIL*)
userGender, userGender': ℙ *(USERID* × *USERGENDER)*
userPhone, userPhone' : ℙ *(USERID* ×*USERPHONE)*
userPassword, userPassword', : ℙ *(USERID* ×*PASSWORD)*
userHistory, userHistory' : ℙ *(USERID* ×*USERHISTORY)*
userStatus, userStatus': ℙ *(USERID* × *USERSTATUS)*
paymentId, paymentId' : ℙ *PAYMENTID*
paymentStatus, paymentStatus': ℙ *(PAYMENTID* ⤔ *PAYMENTSTATUS)*
paymentMethod, paymentMethod': ℙ *(PAYMENTID* ⤔ *PAYMENTMETHOD)*
paymentDate, paymentDate': ℙ *(PAYMENTID* ⤔ *PAYMENTDATE)*
paymentAmount, paymentAmount': *(PAYMENTID* ⤔ ℕ*)*
delivery, delivery': *(PAYMENTID* ⤔ *DELIVERYADDRESS)*
paymentUserId, paymentUserId': *(PAYMENTID* ⤔ *USERID)*
paymentCarId, paymentCartId': *(PAYMENTID* ⤔ *CARTID)*
paymentProductId, paymentProductId': *(PAYMENTID* ⤔ *PRODUCTID)*
cartId, cartId': ℙ *CARTID*
cartProductId, cartProductId': ℙ *(CARTID* ⇸ *PRODID)*
cartUserId, cartUserId': ℙ *(CARTID* ⇸ *USERID)*
voucherID, voucherID': ℙ *(CARTPRODUCTID* ⇸ *VOUCHERID)*
shopeeCoin, shopeeCoin': *(CARTUSERID* ⇸ *SHOPEECOIN)*
itemQty, itemQty': *(CARTPRODUCTID* ⇸ ℕ*)*
subTotal, subTotal': *(CARTPRODUCTID* ⇸ *SUBTOTAL)*
total, total': *(CARTID* ⇸ ℕ*)*
prodId, prodId': ℙ *PRODUCTID*
prodName, prodName': ℙ *(PRODUCTID* × *PRODUCTNAME)*
prodDetail, prodDetail': ℙ *(PRODUCTID* × *PRODUCTDETAIL)*
prodStock, prodStock': ℙ *(PRODUCTID* × *PRODUCTSTOCK)*
prodPrice, prodPrice': ℙ *(PRODUCTID* × *PRODUCTPRICE)*
prodCategory, prodCategory': ℙ*(PRODUCTID* × *PRODUCTCATEGORY)*
user? : *USERID*
paid? : PAYMENTID
cartID! : CARTID
productID! : PRODUCTID
prodName! : PRODUCTNAME

$prodDetail! : PRODUCTDETAIL$

$prodCategory! : PRODUCTCATEGORY$

$status! : PAYMENTSTATUS$

$amount! : PAYMENTAMOUNT$

$date! : PAYMENTDATE$

$rep!: REPORT$

$((user? \in userId \land userStatus(\!|\{user?\}|\!) = loggedIn) \land (paid? \in paymentUserId\sim(\!|\{user?\}|\!) \land paid? \in \text{dom}\ paymentCartId) \land cartID! = paymentCartId(\!|\{paid?\}|\!) \land productID! = cartProductId \rhd \{ paymentCartId (\!|\{paid?\}|\!) \land prodName! = prodName (\!|\{ cartProductId \rhd \{ paymentCartId (\!|\{paid?\}|\!) \}|\!) \land prodDetail! = prodDetail (\!|\{ cartProductId \rhd \{ paymentCartId (\!|\{paid?\}|\!) \}|\!) \land prodCategory! = prodCategory (\!|\{ cartProductId \rhd \{ paymentCartId (\!|\{paid?\}|\!) \}|\!) \land status! = paymentStatus(paid?) \land amount! = paymentAmount(paid?) \land date! = paymentDate(paid?) \land rep! = OKMesssage$

$\land \text{dom}\ userStatus = userId \land \text{dom}\ userStatus' = userId' \land \text{dom}\ paymentStatus = \text{dom}\ paymentDate = \text{dom}\ paymentAmount = dom\ paymentUserId = dom\ paymentCartId = paymentId \land \text{dom}\ paymentStatus' = \text{dom}\ paymentDate' = \text{dom}\ paymentAmount' = dom\ paymentUserId' = dom\ paymentCartId' = paymentId' \land dom\ cartProductId = cartId \land dom\ cartProductId' = cartId' \land \text{dom}\ prodName = \text{dom}\ prodDetail = \text{dom}\ prodCategory = prodId \land \text{dom}\ prodName' = \text{dom}\ prodDetail' = \text{dom}\ prodCategory' = prodId')$

$\lor$

$($

$((user \notin userId \land rep! = UserNotExist \land userId \land userId')$

$\lor$

$(userStatus(\!|\{user?)\}|\!) = loggedOut \land rep! = UserNotLogin \land \text{dom}\ userStatus = userId \land \text{dom}\ userStatus' = userId')$

$\lor$

$(paid? \notin UserID(\!|\{user?\}|\!) \land rep! = PaymentNotExist \land \text{dom}\ paymentUserId = paymentId \land \text{dom}\ paymentUserId' = paymentId'))$

$\land$

$(userId' = userId \land userStatus' = userStatus \land paymentUserId' = paymentUserId \land paymentId' = paymentId))$

4.6 Conclusion

To encapsulate, I'd like to convey my heartfelt appreciation to two amazing staff who have played critical roles in my journey through BACS2083 Formal Methods for Software Engineering. Ms. Mazlinda and Ms. Azurawati have been not only my teachers but also my guiding lights throughout this difficult course. Their enduring commitment to the achievement of their students has left an unforgettable impression on my academic career. Their knowledge of the complex subject matter, which includes Data Abstraction, State Space Schema, Initial Schema, Operation Schema, Error Scenario, and Complete Schema, has been nothing short of impressive. Her ability to explain complex ideas clearly and patiently has been useful to me. I am grateful for their guidance and encouragement, and the lessons they have taught me will stay with me throughout my academic and professional careers.

5.0 Product Management Module

5.1 State Space Schema

Shop

$shopId: \mathbb{P}\ SHOPID$
$shopName: SHOPID \pfun SHOPNAME$
$shopDesc: SHOPID \pfun SHOPDESC$
$shopAdd: SHOPID \pfun SHOPADDRESS$
$shopRating: SHOPID \pfun SHOPRATING$
$shopEmail: SHOPID \pfun SHOPEMAIL$

$shopId = \text{dom}\ shopName = \text{dom}\ shopDesc = \text{dom}\ shopAdd = \text{dom}\ shopRating$

$\#shopRating \leqslant maxRating$

Product

$prodOwner: SHOPID \pfun PRODUCTID$
$prodId: \mathbb{P}\ PRODUCTID$
$prodName: PRODUCTID \pfun PRODUCTNAME$
$prodDetail: PRODUCTID \pfun PRODUCTDETAIL$
$prodStock: PRODUCTID \pfun PRODUCTSTOCK$
$prodPrice: PRODUCTID \pfun PRODUCTPRICE$
$prodCategotry: PRODUCTID \pfun PRODUCTCATEGOTRY$
$prodCondition: PRODUCTID \pfun PRODCONDITION$
$prodBrand: PRODUCTID \pfun PRODUCTBRAND$
$prodWeight: PRODUCTID \pfun PRODUCTWEIGHT$
$prodSize: PRODUCTID \pfun PRODUCTSIZE$
$deliveryFee: PRODUCTID \pfun DELIVERYFEE$
$prodStatus: PRODUCTID \pfun PRODUCTSTATUS$

$prodId = \text{ran}\ prodOwner = \text{dom}\ prodName = \text{dom}\ prodDetail = \text{dom}\ prodStock$
$= \text{dom}\ prodPrice = \text{dom}\ prodCategotry = \text{dom}\ prodCategory = \text{dom}\ prodCondition$
$= \text{dom}\ productBrand = \text{dom}\ prodWeight = \text{dom}\ prodSize = \text{dom}\ deliveryFee = \text{dom}\ productStatus$

$\text{dom}\ prodOwner = shopId$

$prodStock \leqslant maxQuantity$

Voucher

voucherId: *VOUCHERID*

voucherName: *VOUCHERID* $\rightarrowtail$ *VOUCHERNAME*

voucherStartDate: *VOUCHERID* $\rightarrowtail$ *VOUCHERSTARTDATE*

voucherEndDate: *VOUCHERID* $\rightarrowtail$ *VOUCHERENDDATE*

voucherStatus = *VOUCHERID* $\rightarrowtail$ *VOUCHERSTATUS*

voucherDiscount:*VOUCHERID* $\rightarrowtail$ *VOUCHERDISCOUNT*

---

voucherId = dom *voucherName* = dom *voucherStartDate* = dom *voucherEndDate* = dom *voucherStatus* = dom *voucherDiscount*

5.2 Initial State Schema

InitShop

Shop

shopId = ∅
shopName = ∅
shopDesc = ∅
shopAdd = ∅
shopEmail = ∅
shopRating = 0

InitProduct

Product

prodId = ∅
prodOwner = ∅
prodName = ∅
prodDetail = ∅
prod*Category* = ∅
prodCondition = ∅
prodBrand = ∅
prodStatus = *invalid*
prodWeight = 0
prodSize = 0
deliveryFee = 0
prodStock = 0
prodPrice = 0

InitVoucher

Voucher

voucherId = ∅
voucherName = ∅
voucherStartDate = ∅
voucherEndDate = ∅
voucherDiscount = 0
voucherStatus = *invalid*

5.3 Operation Schema

Operation 1: Create, Update, Retrieve & Delete Product

AddProduct

$\Delta Product$

$\Xi Shop$

prodOwner?: $\mathbb{P}$ *SHOPID*

prodId? $\mathbb{P}$ *PRODUCTID*

prodName?: $\mathbb{P}$ *PRODUCTNAME*

prodDetail?: $\mathbb{P}$ *PRODUCTDETAIL*

prodCategory?: $\mathbb{P}$ *PRODUCTCATEGORY*

prodCondition?: $\mathbb{P}$ *PRODUCTCONDITION*

prodBrand?: $\mathbb{P}$ *PRODUCTBRAND*

prodStatus?: $\mathbb{P}$ *PRODUCTSTAUTUS*

prodWeight?: $\mathbb{N}$

prodWidth?: $\mathbb{N}$

prodLength?: $\mathbb{N}$

prodHeight?: $\mathbb{N}$

prodSize?: $\mathbb{N}$

deliveryFee?: $\mathbb{N}$

prodStock?: $\mathbb{N}$

prodPrice?: $\mathbb{N}$

---

shopId? $\in$ *shopId*

prodId? $\notin$ *prodId*

prodCategory? $\in$ ran *prodCategory*

prodCondition? $\in$ *prodCondition*

prodBrand? $\in$ ran *prodBrand*

prodStatus? $\in$ *prodStatus*

prodSize? = *prodLength* * *prodWidth* * *prodHeight*

prodStock? $\leqslant$ *maxQuantity*

prodId ′ = *prodId* $\cup$ *productId*?

prodOwner′ = *prodOwner* $\cup$ {*shopId*? $\mapsto$ *prodId*}

prodName ′ = *prodName* $\cup$ {*prodId*? $\mapsto$ *prodName*?}

prodDetail ′ = *prodDetail* $\cup$ { *prodId*? $\mapsto$ *prodDetail*?}

$prodCategory' = prodCategory \cup \{prodId? \mapsto prodCategoty?\}$

$prodCondition' = prodCondition \cup \{prodId? \mapsto prodCondition?\}$

$prodBrand' = prodBrand \cup \{prodId? \mapsto prodBrand?\}$

$prodStatus' = prodStatus \cup \{prodId? \mapsto prodStatus?\}$

$prodWeight' = prodStatus \cup \{prodId? \mapsto prodWeight?\}$

$prodSize' = prodStatus \cup \{prodId? \mapsto prodSize?\}$

$deliveryFee' = prodStatus \cup \{prodId? \mapsto deliveryFee?\}$

$prodStock' = prodStock \cup \{prodId? \mapsto prodStock?\}$

$prodPrice' = prodPrice \cup \{prodId? \mapsto prodPrice?\}$

RetrieveProduct

$\Xi Product$

$\Xi Shop$

prodName? $\mathbb{P}$ *PRODUCTNAME*

productId! : $\mathbb{P}$ *PRODUCTID*

prodOwner!: $\mathbb{P}$ *SHOPID*

prodDetail!: $\mathbb{P}$ *PRODUCTDETAIL*

prodCategory!: $\mathbb{P}$ *PRODUCTCATEGORY*

prodCondition!: $\mathbb{P}$ *PRODUCTCONDITION*

prodBrand!: $\mathbb{P}$ *PRODUCTBRAND*

prodStatus!: $\mathbb{P}$ *PRODUCTSTAUTUS*

prodWeight!: $\mathbb{N}$

prodSize!: $\mathbb{N}$

deliveryFee!: $\mathbb{N}$

prodStock!: $\mathbb{N}$

prodPrice!: $\mathbb{N}$

---

prodName? $\in$ ran *prodName*

prodId! = dom (*prodName* $\rhd$ {*productName*?}

shopId! = ran (*prodId* $\lhd$ {*prodctId*!})

prodDetail! = *prodDetail* $(\!|\{$ dom (*prodName* $\rhd$ {*productName*?}$\}|\!)$

prodCategory! = *prodCategory* $(\!|\{$ dom (*prodName* $\rhd$ {*productName*?}$\}|\!)$

prodBrand! = *prodBrand* $(\!|\{$ dom (*prodName* $\rhd$ {*productName*?}$\}|\!)$

prodStatus! = *prodStatus* $(\!|\{$ dom (*prodName* $\rhd$ {*productName*?}$\}|\!)$

prodSize! = *prodSize* $(\!|\{$ dom (*prodName* $\rhd$ {*productName*?}$\}|\!)$

prodWeight! = *prodWeight* $(\!|\{$ dom (*prodName* $\rhd$ {*productName*?}$\}|\!)$

deliveryFee! = *deliveryFee* $(\!|\{$ dom (*prodName* $\rhd$ {*productName*?}$\}|\!)$

prodStock! = *prodStock* $(\!|\{$ dom (*prodName* $\rhd$ {*productName*?}$\}|\!)$

prodPrice! = *prodPrice* $(\!|\{$ dom (*prodName* $\rhd$ {*productName*?}$\}|\!)$

UpdateProduct

$\Delta Product$

prodId? : *PRODUCTID*

prodName?: $\mathbb{P}$ *PRODUCTNAME*

prodDetail?: $\mathbb{P}$ *PRODUCTDETAIL*

prodCategory?: $\mathbb{P}$ *PRODUCTCATEGORY*

prodCondition?: $\mathbb{P}$ *PRODUCTCONDITION*

prodBrand?: $\mathbb{P}$ *PRODUCTBRAND*

prodStatus?: $\mathbb{P}$ *PRODUCTSTAUTUS*

prodWeight?: $\mathbb{N}$

prodWidth?: $\mathbb{N}$

prodLength?: $\mathbb{N}$

prodHeight?: $\mathbb{N}$

deliveryFee?: $\mathbb{N}$

prodStock?: $\mathbb{N}$

prodPrice?: $\mathbb{N}$

---

prodId? $\in$ *prodId*

prodCategory? $\in$ ran *prodCategory*

prodBrand? $\in$ ran *prodBrand*

prodStock? $\leqslant$ *maxQuantity*

prodName′ = prodName $\oplus$ {*productId*? $\mapsto$ *productName*?}

$\vee$

prodDetail′ = prodDetail $\oplus$ {*productId*? $\mapsto$ *productDetail*?}

$\vee$

prodCategory′ = prodCategory $\oplus$ {*productId*? $\mapsto$ *prodCategory*?}

$\vee$

prodCondition′ = prodCondition $\oplus$ {*productId*? $\mapsto$ *prodCondition*?}

$\vee$

prodBrand′ = prodBrand $\oplus$ {*productId*? $\mapsto$ *prodBrand*?}

$\vee$

prodStatus′ = prodStatus $\oplus$ {*productId*? $\mapsto$ *prodStatus*?}

$\vee$

prodWeight′ = prodWeight $\oplus$ {*productId*? $\mapsto$ *prodWeight*?}

$\vee$

$$
\begin{aligned}
& prodSize' = prodSize \oplus \{productId? \mapsto (prodLength * prodWeight * prodHeight)\} \\
& \lor \\
& deliveryFee' = deliveryFee \oplus \{productId? \mapsto deliveryFee?\} \\
& \lor \\
& prodStock' = prodStock \oplus \{productId? \mapsto productStock?\} \\
& \lor \\
& prodPrice' = prodPrice \oplus \{productId? \mapsto productPrice?\}
\end{aligned}
$$

DeleteProduct

$\Delta Product$

$prodId? \mathbb{P} PRODUCTID$

$prodId? \in prodId$

$prodId' = prodId \setminus \{productId?\}$

$prodOwner' = prodOwner \rhd \{productId?\}$

$prodName' = \{productId?\} \lhd prodName$

$prodDetail' = \{productId?\} \lhd prodDetail$

$prodCategory' = \{productId?\} \lhd prodCategory$

$prodCondition' = \{productId?\} \lhd prodCondition$

$prodBrand' = \{productId?\} \lhd prodBrand$

$prodStatus' = \{productId?\} \lhd prodStock$

$prodWeight' = \{productId?\} \lhd prodWeight$

$prodSize' = \{productId?\} \lhd prodSize$

$deliveryFee' = \{productId?\} \lhd deliveryFee$

$prodStock' = \{productId?\} \lhd prodStock$

$prodPrice' = \{productId?\} \lhd prodPrice$

Operation 2: Create, Update, Retrieve & Delete Voucher

AddVoucher

$\Delta Voucher$

voucherId? : $\mathbb{P}$ *VOUCHERID*

voucherName?: $\mathbb{P}$ *VOUCHERNAME*

voucherStartDate? : $\mathbb{P}$ *VOUCHERSTARTDATE*

voucherEndDate?: $\mathbb{P}$ *VOUCHERENDDATE*

voucherDiscount?: $\mathbb{N}$

voucherStatus?: $\mathbb{P}$ *VOUCHERSTATUS*

---

voucherId? $\notin$ *voucherId*

voucherId′ = *voucherId* $\cup$ {*voucherId*?}

voucherName′ = *voucherName* $\cup$ {*voucherId*? $\mapsto$ *voucherName*?}

voucherStartDate′ = *voucherStartDate* $\cup$ { *voucherId*? $\mapsto$ *voucherStartDate*?}

voucherEndDate′ = *voucherEndDate* $\cup$ { *voucherId*? $\mapsto$ *voucherEndDate*?}

voucherDiscount′ = *voucherDiscount* $\cup$ {*voucherId*? $\mapsto$ *couherDiscount*?}

voucherStatus′ = *voucherStatus* $\oplus$ { *voucherId*? $\rhd$ *voucherStatus* }

UpdateVoucher

$\Delta Voucher$

voucherId?: $\mathbb{P}$ *VOUCHERID*

voucherName?: $\mathbb{P}$ *VOUCHERNAME*

voucherStartDate? : $\mathbb{P}$ *VOUCHERSTARTDATE*

voucherEndDate?: $\mathbb{P}$ *VOUCHERENDDATE*

voucherDiscount?: $\mathbb{N}$

voucherStatus?: *VOUCHERSTATUS*

---

voucherId? $\in$ *voucherId*

voucherName' = *voucherName* $\oplus$ {*voucherId*? $\mapsto$ *voucherName*?}

$\vee$

voucherStartDate' = *voucherStartDate* $\oplus$ {*voucherId*? $\mapsto$ *voucherStartDate*?}

$\vee$

voucherEndDate' = *voucherEndDate* $\oplus$ {*voucherId*? $\mapsto$ *voucherEndDate*?}

$\vee$

voucherStatus = *voucherStatus* $\oplus$ {*voucherId*? $\mapsto$ *voucherStatus*?}

$\vee$

voucherDiscount = *voucherDiscount* $\oplus$ {*voucherId*? $\mapsto$ *voucherDiscount*?}

RetrieveVoucher

Ξ*Voucher*

voucherName?: $\mathbb{P}$ *VOUCHERNAME*

voucherId!: $\mathbb{P}$ *VOUCHERID*

voucherStartDate!: $\mathbb{P}$ *VOUCHERSTARTDATE*

voucherEndDate!: $\mathbb{P}$ *VOUCHERENDDATE*

voucherDiscount!: $\mathbb{N}$

voucherStatus!: *VOUCHERSTATUS*

---

voucherName? $\in$ ran *voucherName*

voucherId! = dom (*voucherName* $\rhd$ {*voucherName*?})

voucherName! = *voucherName* $(\!|$ { dom (*voucherName* $\rhd$ {*voucherName*?}) } $|\!)$

voucherStartDate! = *voucherStartDate* $(\!|$ { dom (*voucherName* $\rhd$ {*voucherName*?}) } $|\!)$

voucherEndDate! = *voucherEndDate* $(\!|$ { dom (*voucherName* $\rhd$ {*voucherName*?}) } $|\!)$

voucherDiscount! = *voucherDiscount* $(\!|$ { dom (*voucherName* $\rhd$ {*voucherName*?})} $|\!)$

DeleteVoucherByName

Δ*Voucher*

voucherId?: $\mathbb{P}$ *VOUCHERID*

---

voucherId? $\in$ *voucherId*

voucherId′ = *voucherId* $\setminus$ *voucherId*?

voucherName′ = {*voucherId*?} $\lhd$ *voucherName*

voucherStartDate′ = {*voucherId*?} $\lhd$ *voucherStartDate*

voucherEndDate′ = {*voucherId*?} $\lhd$ *voucherEndDate*

voucherDiscount′ = {*voucherId*?} $\lhd$ *voucherDiscount*

voucherStatus′ = {*voucherId*?} $\lhd$ *voucherStatus*

Operation 3: Create, Update, Retrieve & Delete Shop

AddShop

$\Delta Shop$

$shopId? : \mathbb{P}\ SHOPID$

$shopName? : \mathbb{P}\ SHOPNAME$

$shopDesc? : \mathbb{P} SHOPDESC$

$shopAdd? : \mathbb{P}\ SHOPADDRESS$

$shopEmail? \ \mathbb{P}\ SHOPEMAIL$

$shopId? \notin shopId$

$shopId' = shopId \cup shopId?$

$shopName' = shopName \cup \{shopId? \mapsto shopName?\}$

$shopDesc' = shopDesc \cup \{shopId? \mapsto shopDesc?\}$

$shopAdd' = shopAdd \cup \{shopId? \mapsto shopAdd?\}$

$shopEmail' = shopEmail \cup \{shopId? \mapsto shopEmail?\}$

UpdateShopName

$\Delta Shop$

$shopId? : \mathbb{P}\ SHOPID$

$shopName? : \mathbb{P}\ SHOPNAME$

$shopDesc? : \mathbb{P} SHOPDESC$

$shopAdd? : \mathbb{P}\ SHOPADDRESS$

$shopEmail? : \mathbb{P}\ SHOPEMAIL$

$shopRating? : \mathbb{N}$

$shopId? \in shopId$

$shopName' = shopName \oplus \{shopId? \mapsto shopName?\}$

$\vee$

$shopDesc' = shopDesc \oplus \{shopId? \mapsto shopDesc?\}$

$\vee$

$shopAdd' = shopAdd \oplus \{shopId? \mapsto shopAdd?\}$

$\vee$

$shopEmail' = shopEmail \oplus \{shopId? \mapsto shopEmail?\}$

RetrieveShop

$\Xi Shop$

shopName?: *SHOPNAME*

shopId!: *SHOPID*

shopDesc!: *SHOPDESC*

shopAdd!: *SHOPADDRESS*

shopRating!: *SHOPRATING*

shopEmail!: *SHOPEMAIL*

---

shopName? $\in$ ran *shopName*

shopId! = dom (*shopName* $\rhd$ {*shopName*?})

shopDesc! = *shopDesc* $(\!|$ {dom (*shopName* $\rhd$ {*shopName*?})} $|\!)$

shopAdd! = *shopAdd* $(\!|$ { dom (*shopName* $\rhd$ {*shopName*?})} $|\!)$

shopRating! = *shopAdd* $(\!|$ { dom (*shopName* $\rhd$ {*shopName*?})} $|\!)$

shopeEmail! = *shopAdd* $(\!|$ { dom (*shopName* $\rhd$ {*shopName*?})} $|\!)$

DeleteShop

$\Delta Shop$

shopId?: *SHOPID*

---

shopId? $\in$ *shopId*

shopId′ = *shopId* $\setminus$ *shopId*?

shopName′ = {*shopId*?} $\lhd$ *shopName*

shopDesc′ = {*shopId*?} $\lhd$ *shopDesc*

shopAdd′ = {*shopId*?} $\lhd$ *shopAdd*

shopRating′ = {*shopId*?} $\lhd$ *shopRating*

5.4 Error Scenarios

Error Scenario Table

Schema Name	Success Pre-Condition	Failure Pre-Condition	Remark
AddProduct	prodId? ∉ prodId prodCategory? ∈ prodCategory shopId? ∈ shopId prodBrand? ∈ prodBrand prodStock? ≤ maxQuantity	prodId? ∈ prodId prodCategory ∉ prodCategory shopId? ∈ shopId prodStock? > maxQuantity	Product Existed Category Not Existing Shop Not Existing Brand Not Existing Invalid Quantity
RetriveProduct	prodName? ∈ prodName	prodName? ∉ prodName	Product Name Not Existing
UpdateProduct	prodId? ∈ prodId prodCategory? ∈ prodCategory prodBrand? ∈ prodBrand prodStock ⩽ maxQuantity	prodId? ∉ prodId prodCategory? ∉ prodCategory prodBrand? ∉ prodBrand prodStock > maxQuantity	Product Not Existing Category not existing Brand Not Existing Invalid Quantity
DeleteProduct	prodId? ∈ prodId	prodId? ∉ prodId	Product Not Existing
AddVoucher	voucherId? ∉ voucherId	voucherId? ∈ voucherId	Voucher Not Existing
RetrieveVoucher	voucherName? ∈ voucherName	voucherName? ∉ voucherName	Voucher Name Not Existing
UpdateVoucher	voucherId? ∈ voucherId	voucherId? ∉ voucherId	Voucher Not Existing
DeleteVoucher	voucherId? ∈ voucherId	voucherId? ∉ voucherId	Voucher Not Existing
AddShop	shopId? ∉ shopId shopEmail? ∉ shopEmail	shopId? ∈ shopId shopEmail? ∈ shopEmail	Shop Existing Email Existed
RetrieveShop	shopName? ∈ shopName	shopName? ∉ shopName	Shop Not Existing

UpdateShop	shopId? ∈ shopId shopEmail? ∉ shopEmail	shopId? ∉ shopId shopEmail? ∈ shopEmail	Shop Not Existing Email Existed
DeleteShop	shopId? ∈ shopId	shopId? ∉ shopId	Shop Not Existing

Error Scenario Free Type

RESPONSE ::= success | productNotExist | productExist | categoryNotExist | invalidQuantity | brandNotExist |voucherNotExist | voucherExist | shopNotExist | shopExist | emailExist

Okay

$rep! : RESPONSE$

$rep! = success$

Error Scenario

AddProductError

$\Xi Product$
$\Xi Shop$

$prodId? : PRODUCTID$
$prodCategory? : PRODUCTCATEGORY$
$prodBrand? : PRODUCTBRAND$
$shopId? : SHOPID$
$prodStock? : \mathbb{N}$
$rep! : RESPONSE$

$(prodId? \in prodId \land rep! = productExist)$
$\lor$
$(\#prodStock? > maxQuantity \land rep! = invalidQuantity)$
$\lor$
$(prodCategory? \notin prodCategory \land rep! = categoryNotExist)$
$\lor$
$(prodBrand? \notin prodBrand \land rep! = brandNotExist)$
$\lor$
$(shopId? \notin shopId \land rep! = shopNotExist)$

RetriveProductError

$\Xi Product$

$prodId? : PRODUCTID$
$rep! : RESPONSE$

$prodId? \notin prodId \land rep! = productNotExist$

UpdateProductError

$\Xi Product$

$prodId?: PRODUCTID$
$prodCategory?: PRODUCTCATEGORY$
$prodBrand?: PRODCUTBRAND$

$prodStock?: \mathbb{N}$
$rep!: RESPONSE$

$(prodId? \in prodId \land rep! = productExist)$
$\lor$
$(\#prodStock? > maxQuantity \land rep! = invalidQuantity)$
$\lor$
$(prodCategory? \notin prodCategory \land rep! = categoryNotExist)$
$\lor$
$(prodBrand? \notin prodBrand \land rep! = brandNotExist)$

DeleteProductError

$\Xi Product$

$prodId?: PRODUCTID$
$rep!: RESPONSE$

$prodId? \notin prodId \land rep! = productNotExist$

AddVoucherError

$\Xi Product$

$prodId?: PRODUCTID$
$rep!: RESPONSE$

$voucherId? \in voucherId \land rep! = voucherExist$

RetriveVoucherError

$\Xi Product$

prodId?: *PRODUCTID*
rep!: *RESPONSE*

---

voucherId? $\notin$ *voucherId* $\land$ *rep*! = *voucherNotExist*

UpdateVoucherError

$\Xi Product$

prodId?: *PRODUCTID*
rep!: *RESPONSE*

---

voucherId? $\notin$ *voucherId* $\land$ *rep*! = *voucherNotExist*

DeleteVoucherError

$\Xi Product$

prodId?: *PRODUCTID*
rep!: *RESPONSE*

---

voucherId? $\notin$ *voucherId* $\land$ *rep*! = *voucherNotExist*

AddShopError

$\Xi Shop$

shopId?: *SHOPID*
shopEmail?: *SHOPEMAIL*
rep!: *RESPONSE*

(*shopId*? $\in$ *shopId* $\land$ *rep*! = *shopExist*)
$\lor$
(*shopEmail*? $\in$ *shopEmail* $\land$ *rep*! = *emailExist*)

RetieveShopError

$\Xi Shop$

shopId?: *SHOPID*
rep!: *RESPONSE*

(*shopId*? $\notin$ *shopId* $\land$ *rep*! = *shopNotExist*)

UpdateShopError

$\Xi Shop$

shopId?: *SHOPID*
rep!: *RESPONSE*

---

shopId? $\notin$ *shopId* $\land$ *rep*! = *shopNotExist*
$\lor$
(*shopEmail*? $\in$ *shopEmail* $\land$ *rep*! = *emailNotExist*)

DeleteShopError

$\Xi Shop$

shopId?: *SHOPID*
rep!: *RESPONSE*

---

shopId? $\notin$ *shopId* $\land$ *rep*! = *shopNotExist*

5.5 Complete Schema

AddProductComplete ≙ (AddProduct ∧ Okay) ∨ AddProductError

AddProductComplete

$prodOwner, prodOwner' : SHOPID \pfun PRODUCTID$
$prodId, prodId' : \mathbb{P}\ PRODUCTID$
$prodName, prodName` : PRODUCTID \pfun PRODUCTNAME$
$prodDetail, prodDetail` : PRODUCTID \pfun PRODUCTDETAIL$
$prodStock, prodStock` : PRODUCTID \pfun PRODUCTSTOCK$
$prodPrice, prodPrice` : PRODUCTID \pfun PRODUCTPRICE$
$prodCategotry, prodCategotry` : PRODUCTID \pfun PRODUCTCATEGOTRY$
$prodCondition, prodCondition` : PRODUCTID \pfun PRODCONDITION$
$prodBrand, prodBrand` : PRODUCTID \pfun PRODUCTBRAND$
$prodWeight, prodWeight` : PRODUCTID \pfun PRODUCTWEIGHT$
$prodSize, prodSize` : PRODUCTID \pfun PRODUCTSIZE$
$deliveryFee, deliveryFee` : PRODUCTID \pfun DELIVERYFEE$
$prodStatus, prodStatus` : PRODUCTID \pfun PRODUCTSTATUS$

$prodOwner? : \mathbb{P}\ SHOPID$
$prodId? \ \mathbb{P}\ PRODUCTID$
$prodName? : \mathbb{P}\ PRODUCTNAME$
$prodDetail? : \mathbb{P}\ PRODUCTDETAIL$
$prodCategory? : \mathbb{P}\ PRODUCTCATEGORY$
$prodCondition? : \mathbb{P}\ PRODUCTCONDITION$
$prodBrand? : \mathbb{P}\ PRODUCTBRAND$
$prodStatus? : \mathbb{P}\ PRODUCTSTAUTUS$
$prodWeight? : \mathbb{N}$
$prodWidth? : \mathbb{N}$
$prodLength? : \mathbb{N}$
$prodHeight? : \mathbb{N}$
$prodSize? : \mathbb{N}$
$deliveryFee? : \mathbb{N}$
$prodStock? : \mathbb{N}$
$prodPrice? : \mathbb{N}$
$rep! : RESPONSE$

$(shopId? \in shopId \land prodId? \notin prodId \land prodCategory? \in \text{ran}\ prodCategory \land$
$prodCondition? \in prodCondition \land prodBrand? \in \text{ran}\ prodBrand \land prodStatus? \in prodStatus \land$
$prodSize? = prodLength * prodWidth * prodHeight \land prodStock? \leqslant maxQuantity \land rep! : success \land$
$prodId' = prodId \cup productId? \land prodOwner' = prodOwner \cup \{shopId? \mapsto prodId\} \land$

$$
\begin{aligned}
&prodName' = prodName \cup \{prodId? \mapsto prodName?\} \land prodDetail' = prodDetail \cup \{ prodId? \mapsto prodDetail?\} \\
&\land prodCategory' = prodCategory \cup \{prodId? \mapsto prodCategoty?\} \land \\
&prodCondition' = prodCondition \cup \{prodId? \mapsto prodCondition?\} \land \\
&prodBrand' = prodBrand \cup \{prodId? \mapsto prodBrand?\} \land prodStatus' = prodStatus \cup \{prodId? \mapsto prodStatus?\} \land \\
&prodWeight' = prodStatus \cup \{prodId? \mapsto prodWeight?\} \land prodSize' = prodStatus \cup \{prodId? \mapsto prodSize?\} \land \\
&deliveryFee' = prodStatus \cup \{prodId? \mapsto deliveryFee?\} \land prodStock' = prodStock \cup \{prodId? \mapsto prodStock?\} \land \\
&prodPrice' = prodPrice \cup \{prodId? \mapsto prodPrice?\}) \\
&\land \\
&(prodId = \text{ran } prodOwner = \text{dom } prodName = \text{dom } prodDetail = \text{dom } prodStock \\
&= \text{dom } prodPrice = \text{dom } prodCategotry = \text{dom } prodCategory = \text{dom } prodCondition \\
&= \text{dom } productBrand = \text{dom } prodWeight = \text{dom } prodSize = \text{dom } deliveryFee = \text{dom } productStatus \land \\
&prodStock \leqslant maxQuantity \land \text{dom } prodOwner = shopId) \\
&\land \\
&(prodId` = \text{ran } prodOwner` = \text{dom } prodName` = \text{dom } prodDetail` = \text{dom } prodStock` \\
&= \text{dom } prodPrice` = \text{dom } prodCategotry` = \text{dom } prodCategory` = \text{dom } prodCondition` \\
&= \text{dom } productBran`d = \text{dom } prodWeight` = \text{dom } prodSize` = \text{dom } deliveryFee` = \text{dom } productStatus` \land \\
&prodStock` \leqslant maxQuantity \land \text{dom } prodOwner` = shopId`) \\
&\lor \\
&(prodId? \in prodId \land rep! = productExist) \land prodId = \text{ran } prodOwner = \text{dom } prodName = \text{dom } prodDetail = \\
&\text{dom } prodStock = \text{dom } prodPrice = \text{dom } prodCategotry = \text{dom } prodCategory = \text{dom } prodCondition \\
&= \text{dom } productBrand = \text{dom } prodWeight = \text{dom } prodSize = \text{dom } deliveryFee = \text{dom } productStatus \land \\
&prodId` = \text{ran } prodOwner` = \text{dom } prodName` = \text{dom } prodDetail` = \text{dom } prodStock` \\
&= \text{dom } prodPrice` = \text{dom } prodCategotry` = \text{dom } prodCategory` = \text{dom } prodCondition` \\
&= \text{dom } productBran`d = \text{dom } prodWeight` = \text{dom } prodSize` = \text{dom } deliveryFee` = \text{dom } productStatus`) \\
&\lor \\
&(\#prodStock? > maxQuantity \land rep! = invalidQuantity) \land prodStock \leqslant maxQuantity \land \\
&prodStock` \leqslant maxQuantity) \\
&\lor \\
&(prodCategory? \notin prodCategory \land rep! = categoryNotExist) \land (\text{dom } prodCategory = prodId = \\
&\text{ran } prodOwner = \text{dom } prodName = \text{dom } prodDetail = \text{dom } prodStock \\
&= \text{dom } prodPrice = \text{dom } prodCategotry = \text{dom } prodCondition \\
&= \text{dom } productBrand = \text{dom } prodWeight = \text{dom } prodSize = \text{dom } deliveryFee = \text{dom } productStatus \\
&\land prodStock \leqslant maxQuantity) \land \text{dom } prodCategory` = prodId` = \text{ran } prodOwner` = \text{dom } prodName` = \\
&\text{dom } prodDetail` = \text{dom } prodStock` \\
&= \text{dom } prodPrice` = \text{dom } prodCategotry` = \text{dom } prodCondition` \\
&= \text{dom } productBran`d = \text{dom } prodWeight` = \text{dom } prodSize` = \text{dom } deliveryFee` = \text{dom } productStatus`) \\
&\lor \\
&(prodBrand? \notin prodBrand \land rep! = brandNotExist) \land (\text{dom } productBrand = prodId = \text{ran } prodOwner \\
&= \text{dom } prodName = \text{dom } prodDetail = \text{dom } prodStock \\
&= \text{dom } prodPrice = \text{dom } prodCategotry = \text{dom } prodCategory = \text{dom } prodCondition \\
&= \text{dom } prodWeight = \text{dom } prodSize = \text{dom } deliveryFee = \text{dom } productStatus
\end{aligned}
$$

$$
\begin{array}{l}
\land\ prodStock \leqslant maxQuantity) \\
\land \\
(\mathrm{dom}\ prodCategotry` = prodId` = \mathrm{ran}\ prodOwner` = \mathrm{dom}\ prodName` = \mathrm{dom}\ prodDetail` = \mathrm{dom}\ prodStock` \\
= \mathrm{dom}\ prodPrice` = \mathrm{dom}\ prodCategory` = \mathrm{dom}\ prodCondition` \\
= \mathrm{dom}\ productBran`d = \mathrm{dom}\ prodWeight` = \mathrm{dom}\ prodSize` = \mathrm{dom}\ deliveryFee` = \mathrm{dom}\ productStatus`) \\
\lor \\
(shopId? \notin shopId \land rep! = shopNotExist) \land \mathrm{dom}\ prodOwner = shopId \land \mathrm{dom}\ prodOwner` = shopId`
\end{array}
$$

RetrieveProductComplete ≙ (RetrieveProduct ∧ Okay) ∨ RetrieveProductError

RetrieveProductComplete

$prodOwner, prodOwner' : SHOPID \pfun PRODUCTID$
$prodId, prodId' : \mathbb{P}\ PRODUCTID$
$prodName, prodName` : PRODUCTID \pfun PRODUCTNAME$
$prodDetail, prodDetail` : PRODUCTID \pfun PRODUCTDETAIL$
$prodStock, prodStock` : PRODUCTID \pfun PRODUCTSTOCK$
$prodPrice, prodPrice` : PRODUCTID \pfun PRODUCTPRICE$
$prodCategotry, prodCategotry` : PRODUCTID \pfun PRODUCTCATEGOTRY$
$prodCondition, prodCondition` : PRODUCTID \pfun PRODCONDITION$
$prodBrand, prodBrand` : PRODUCTID \pfun PRODUCTBRAND$
$prodWeight, prodWeight` : PRODUCTID \pfun PRODUCTWEIGHT$
$prodSize, prodSize` : PRODUCTID \pfun PRODUCTSIZE$
$deliveryFee, deliveryFee` : PRODUCTID \pfun DELIVERYFEE$
$prodStatus, prodStatus` : PRODUCTID \pfun PRODUCTSTATUS$

$prodName? \mathbb{P}\ PRODUCTNAME$
$productId! : \mathbb{P}\ PRODUCTID$
$prodOwner! : \mathbb{P}\ SHOPID$
$prodDetail! : \mathbb{P}\ PRODUCTDETAIL$
$prodCategory! : \mathbb{P}\ PRODUCTCATEGORY$
$prodCondition! : \mathbb{P}\ PRODUCTCONDITION$
$prodBrand! : \mathbb{P}\ PRODUCTBRAND$
$prodStatus! : \mathbb{P}\ PRODUCTSTAUTUS$

$prodWeight! : \mathbb{N}$
$prodSize! : \mathbb{N}$
$deliveryFee! : \mathbb{N}$
$prodStock! : \mathbb{N}$
$prodPrice! : \mathbb{N}$

$prodName? \in \text{ran}\ prodName \land$
$prodId! = \text{dom}\ (prodName \rhd \{productName?\} \land shopId! = \text{ran}\ (prodId \lhd \{prodctId!\}) \land$
$prodDetail! = prodDetail\ (\!|\{\ \text{dom}\ (prodName \rhd \{productName?\}\}|\!) \land$
$prodCategory! = prodCategory\ \ (\!|\{\ \text{dom}\ (prodName \rhd \{productName?\}\}|\!)\ \land$
$prodBrand! = prodBrand\ (\!|\{\ \text{dom}\ (prodName \rhd \{productName?\}\}|\!) \land$
$prodStatus! = prodStatus\ (\!|\{\ \text{dom}\ (prodName \rhd \{productName?\}\}|\!) \land$
$prodSize! = prodSize\ (\!|\{\ \text{dom}\ (prodName \rhd \{productName?\}\}|\!) \land$
$prodWeight! = prodWeight\ (\!|\{\ \text{dom}\ (prodName \rhd \{productName?\}\}|\!) \land$
$deliveryFee! = deliveryFee\ (\!|\{\ \text{dom}\ (prodName \rhd \{productName?\}\}|\!) \land$

$$
\begin{aligned}
&prodStock! = prodStock \ (\!|\{ \text{dom } (prodName \rhd \{productName?\}\}|\!) \land \\
&prodPrice! = prodPrice \ (\!|\{ \text{dom } (prodName \rhd \{productName?\}\}|\!) \\
&\land \\
&(prodId = \text{ran } prodOwner = \text{dom } prodName = \text{dom } prodDetail = \text{dom } prodStock \\
&= \text{dom } prodPrice = \text{dom } prodCategotry = \text{dom } prodCategory = \text{dom } prodCondition \\
&= \text{dom } productBrand = \text{dom } prodWeight = \text{dom } prodSize = \text{dom } deliveryFee = \text{dom } productStatus \\
&\land prodStock \leqslant maxQuantity \\
&\land \text{dom } prodOwner = shopId) \\
&\land \\
&(prodId` = \text{ran } prodOwner` = \text{dom } prodName` = \text{dom } prodDetail` = \text{dom } prodStock` \\
&= \text{dom } prodPrice` = \text{dom } prodCategotry` = \text{dom } prodCategory` = \text{dom } prodCondition` \\
&= \text{dom } productBran`d = \text{dom } prodWeight` = \text{dom } prodSize` = \text{dom } deliveryFee` = \text{dom } productStatus` \\
&\land prodStock` \leqslant maxQuantity \\
&\land \text{dom } prodOwner` = shopId`) \\
&\lor \\
&prodId? \notin prodId \land rep! = productNotExist \land (prodId = \text{ran } prodOwner = \text{dom } prodName = \\
&\text{dom } prodDetail = \text{dom } prodStock \\
&= \text{dom } prodPrice = \text{dom } prodCategotry = \text{dom } prodCategory = \text{dom } prodCondition \\
&= \text{dom } productBrand = \text{dom } prodWeight = \text{dom } prodSize = \text{dom } deliveryFee = \text{dom } productStatus) \\
&\land \\
&(prodId` = \text{ran } prodOwner` = \text{dom } prodName` = \text{dom } prodDetail` = \text{dom } prodStock` \\
&= \text{dom } prodPrice` = \text{dom } prodCategotry` = \text{dom } prodCategory` = \text{dom } prodCondition` \\
&= \text{dom } productBran`d = \text{dom } prodWeight` = \text{dom } prodSize` = \text{dom } deliveryFee` = \text{dom } productStatus`) \\
&\land \\
&(prodOwner = prodOwner' \land prodId = prodId' \land prodName = prodName` \land prodDetail = prodDetail` \land \\
&prodStock = prodStock` \land prodPrice = prodPrice` \land prodCategotry = prodCategotry` \land \\
&prodCondition = prodCondition` \land prodBrand = prodBrand \land prodWeight = prodWeight` \land \\
&prodSize = prodSize` \land deliveryFee = deliveryFee \land \\
&prodStatus = prodStatus`)
\end{aligned}
$$

UpdateProductComplete ≙ (UpdateProduct ∧ Okay) ∨ UpdateProductError

UpdateProductComplete

$prodOwner, prodOwner' : SHOPID \pfun PRODUCTID$
$prodId, prodId' : \mathbb{P}\ PRODUCTID$
$prodName, prodName` : PRODUCTID \pfun PRODUCTNAME$
$prodDetail, prodDetail` : PRODUCTID \pfun PRODUCTDETAIL$
$prodStock, prodStock` : PRODUCTID \pfun PRODUCTSTOCK$
$prodPrice, prodPrice` : PRODUCTID \pfun PRODUCTPRICE$
$prodCategotry, prodCategotry` : PRODUCTID \pfun PRODUCTCATEGOTRY$
$prodCondition, prodCondition` : PRODUCTID \pfun PRODCONDITION$
$prodBrand, prodBrand` : PRODUCTID \pfun PRODUCTBRAND$
$prodWeight, prodWeight` : PRODUCTID \pfun PRODUCTWEIGHT$
$prodSize, prodSize` : PRODUCTID \pfun PRODUCTSIZE$
$deliveryFee, deliveryFee` : PRODUCTID \pfun DELIVERYFEE$
$prodStatus, prodStatus` : PRODUCTID \pfun PRODUCTSTATUS$

$prodOwner? : \mathbb{P}\ SHOPID$
$prodId? \ \mathbb{P}\ PRODUCTID$
$prodName? : \mathbb{P}\ PRODUCTNAME$
$prodDetail? : \mathbb{P}\ PRODUCTDETAIL$
$prodCategory? : \mathbb{P}\ PRODUCTCATEGORY$
$prodCondition? : \mathbb{P}\ PRODUCTCONDITION$
$prodBrand? : \mathbb{P}\ PRODUCTBRAND$
$prodStatus? : \mathbb{P}\ PRODUCTSTAUTUS$
$prodWeight? : \mathbb{N}$
$prodWidth? : \mathbb{N}$
$prodLength? : \mathbb{N}$
$prodHeight? : \mathbb{N}$
$prodSize? : \mathbb{N}$
$deliveryFee? : \mathbb{N}$
$prodStock? : \mathbb{N}$
$prodPrice? : \mathbb{N}$
$rep! : RESPONSE$

$prodId? \in prodId$
$\land$
$prodCategory? \in \text{ran}\ prodCategory \land prodBrand? \in \text{ran}\ prodBrand \land prodStock? \leqslant maxQuantity$
$\land$
$(prodName' = prodName \oplus \{productId? \mapsto productName?\} \lor$
$prodDetail' = prodDetail \oplus \{productId? \mapsto productDetail?\}$

$$
\begin{aligned}
&\vee \\
&prodCategory' = prodCategory \oplus \{productId? \mapsto prodCategory?\} \vee \\
&prodCondition' = prodCondition \oplus \{productId? \mapsto prodCondition?\} \\
&\vee \\
&prodBrand' = prodBrand \oplus \{productId? \mapsto prodBrand?\} \vee \\
&prodStatus' = prodStatus \oplus \{productId? \mapsto prodStatus?\} \\
&\vee \\
&prodWeight' = prodWeight \oplus \{productId? \mapsto prodWeight?\} \vee \\
&prodSize' = prodSize \oplus \{productId? \mapsto (prodLength * prodWeight * prodHeight)\} \\
&\vee \\
&deliveryFee' = deliveryFee \oplus \{productId? \mapsto deliveryFee?\} \vee \\
&prodStock' = prodStock \oplus \{productId? \mapsto productStock?\} \\
&\vee \\
&prodPrice' = prodPrice \oplus \{productId? \mapsto productPrice?\}) \\
&\wedge \\
&(prodId = \text{ran } prodOwner = \text{dom } prodName = \text{dom } prodDetail = \text{dom } prodStock \\
&= \text{dom } prodPrice = \text{dom } prodCategotry = \text{dom } prodCategory = \text{dom } prodCondition \\
&= \text{dom } productBrand = \text{dom } prodWeight = \text{dom } prodSize = \text{dom } deliveryFee = \text{dom } productStatus \wedge \\
&\ prodStock \leqslant maxQuantity \wedge \text{dom } prodOwner = shopId) \\
&\wedge \\
&(prodId` = \text{ran } prodOwner` = \text{dom } prodName` = \text{dom } prodDetail` = \text{dom } prodStock` \\
&= \text{dom } prodPrice` = \text{dom } prodCategotry` = \text{dom } prodCategory` = \text{dom } prodCondition` \\
&= \text{dom } productBran`d = \text{dom } prodWeight` = \text{dom } prodSize` = \text{dom } deliveryFee` = \text{dom } productStatus` \wedge \\
&prodStock` \leqslant maxQuantity \wedge \text{dom } prodOwner` = shopId`) \\
&\vee \\
&prodId? \notin prodId \wedge rep! = productNotExist \wedge (prodId = \text{ran } prodOwner = \text{dom } prodName = \\
&\text{dom } prodDetail = \text{dom } prodStock \\
&= \text{dom } prodPrice = \text{dom } prodCategotry = \text{dom } prodCategory = \text{dom } prodCondition \\
&= \text{dom } productBrand = \text{dom } prodWeight = \text{dom } prodSize = \text{dom } deliveryFee = \text{dom } productStatus) \\
&\wedge \\
&(prodId` = \text{ran } prodOwner` = \text{dom } prodName` = \text{dom } prodDetail` = \text{dom } prodStock` \\
&= \text{dom } prodPrice` = \text{dom } prodCategotry` = \text{dom } prodCategory` = \text{dom } prodCondition` \\
&= \text{dom } productBran`d = \text{dom } prodWeight` = \text{dom } prodSize` = \text{dom } deliveryFee` = \text{dom } productStatus`)
\end{aligned}
$$

DeleteProductComplete ≙ (DeleteProduct ∧ Okay) ∨ DeleteProductError

DeleteProductComplete

$prodOwner, prodOwner' : SHOPID \pfun PRODUCTID$
$prodId, prodId' : \mathbb{P}\ PRODUCTID$
$prodName, prodName` : PRODUCTID \pfun PRODUCTNAME$
$prodDetail, prodDetail` : PRODUCTID \pfun PRODUCTDETAIL$
$prodStock, prodStock` : PRODUCTID \pfun PRODUCTSTOCK$
$prodPrice, prodPrice` : PRODUCTID \pfun PRODUCTPRICE$
$prodCategotry, prodCategotry` : PRODUCTID \pfun PRODUCTCATEGOTRY$
$prodCondition, prodCondition` : PRODUCTID \pfun PRODCONDITION$
$prodBrand, prodBrand` : PRODUCTID \pfun PRODUCTBRAND$
$prodWeight, prodWeight` : PRODUCTID \pfun PRODUCTWEIGHT$
$prodSize, prodSize` : PRODUCTID \pfun PRODUCTSIZE$
$deliveryFee, deliveryFee` : PRODUCTID \pfun DELIVERYFEE$
$prodStatus, prodStatus` : PRODUCTID \pfun PRODUCTSTATUS$

$prodId? \mathbb{P}\ PRODUCTID$
$rep! : RESPONSE$

$prodId? \in prodId$
$\land$
$prodId' = prodId \setminus \{productId?\} \land prodOwner' = prodOwner \rhd \{productId?\} \land$
$prodName' = \{productId?\} \lhd prodName \land$
$prodDetail' = \{productId?\} \lhd prodDetail \land prodCategory' = \{productId?\} \lhd prodCategory \land$
$prodCondition' = \{productId?\} \lhd prodCondition \land$
$prodBrand' = \{productId?\} \lhd prodBrand \land prodStatus' = \{productId?\} \lhd prodStock \land$
$prodWeight' = \{productId?\} \lhd prodWeight \land$
$prodSize' = \{productId?\} \lhd prodSize \land deliveryFee' = \{productId?\} \lhd deliveryFee \land$
$prodStock' = \{productId?\} \lhd prodStock \land$
$prodPrice' = \{productId?\} \lhd prodPrice$
$\land$
($prodId$ = ran $prodOwner$ =dom $prodName$ = dom $prodDetail$ = dom $prodStock$
= dom $prodPrice$ = dom $prodCategotry$ = dom $prodCategory$ = dom $prodCondition$
= dom $productBrand$ = dom $prodWeight$ = dom $prodSize$ = dom $deliveryFee$ = dom $productStatus$ $\land$
$prodStock \leqslant maxQuantity \land$ dom $prodOwner = shopId$)
$\land$
($prodId$` = ran $prodOwner$` =dom $prodName$` = dom $prodDetail$` = dom $prodStock$`
= dom $prodPrice$` = dom $prodCategotry$` = dom $prodCategory$` = dom $prodCondition$`
= dom $productBran$`$d$ = dom $prodWeight$` = dom $prodSize$` = dom $deliveryFee$` = dom $productStatus$`

$$
\begin{aligned}
&\wedge\ prodStock` \leqslant maxQuantity \wedge \mathrm{dom}\ prodOwner` = shopId`) \\
&\vee \\
&prodId? \notin prodId \wedge rep! = productNotExist \wedge (prodId = \mathrm{ran}\ prodOwner = \mathrm{dom}\ prodName = \\
&\mathrm{dom}\ prodDetail = \mathrm{dom}\ prodStock \\
&= \mathrm{dom}\ prodPrice = \mathrm{dom}\ prodCategotry = \mathrm{dom}\ prodCategory = \mathrm{dom}\ prodCondition \\
&= \mathrm{dom}\ productBrand = \mathrm{dom}\ prodWeight = \mathrm{dom}\ prodSize = \mathrm{dom}\ deliveryFee = \mathrm{dom}\ productStatus) \\
&\wedge \\
&(prodId` = \mathrm{ran}\ prodOwner` = \mathrm{dom}\ prodName` = \mathrm{dom}\ prodDetail` = \mathrm{dom}\ prodStock` \\
&= \mathrm{dom}\ prodPrice` = \mathrm{dom}\ prodCategotry` = \mathrm{dom}\ prodCategory` = \mathrm{dom}\ prodCondition` \\
&= \mathrm{dom}\ productBran`d = \mathrm{dom}\ prodWeight` = \mathrm{dom}\ prodSize` = \mathrm{dom}\ deliveryFee` = \mathrm{dom}\ productStatus`)
\end{aligned}
$$

AddVoucherComplete ≙ (AddVoucher ∧ Okay) ∨ AddVoucherError

AddVoucherComplete

voucherId,voucherId`: *VOUCHERID*
voucherName,voucherName`: *VOUCHERID* ⇸ *VOUCHERNAME*
voucherStartDate,voucherStarDate`: *VOUCHERID* ⇸*VOUCHERSTARTDATE*
voucherEndDate,voucherEndDate`: *VOUCHERID* ⇸ *VOUCHERENDDATE*
voucherStatus,voucherStatus` = *VOUCHERID* ⇸ *VOUCHERSTATUS*
voucherDiscount,voucherDiscount`:*VOUCHERID* ⇸ *VOUCHERDISCOUNT*

voucherId? : ℙ *VOUCHERID*
voucherName?: ℙ *VOUCHERNAME*
voucherStartDate? : ℙ *VOUCHERSTARTDATE*
voucherEndDate?: ℙ *VOUCHERENDDATE*
voucherDiscount?: ℕ
voucherStatus?: ℙ *VOUCHERSTATUS*
rep!: *RESPONSE*

---

voucherId? ∉ *voucherId*
∧
voucherId′ = *voucherId* ∪ {*voucherId*?} ∧ *voucherName′* = *voucherName* ∪ {*voucherId*? ↦ *voucherName*?} ∧
voucherStartDate′ = *voucherStartDate* ∪ { *voucherId*? ↦ *voucherStartDate*?} ∧
voucherEndDate′ = *voucherEndDate* ∪ { *voucherId*? ↦ *voucherEndDate*?} ∧
voucherDiscount′ = *voucherDiscount* ∪ {*voucherId*? ↦ *couherDiscount*?} ∧
voucherStatus′ = *voucherStatus* ⊕ { *voucherId*? ⊳ *voucherStatus* }
∧
voucherId = dom *voucherName* = dom *voucherStartDate* = dom *voucherEndDate*
= dom *voucherStatus* = dom *voucherDiscount*
∧
voucherId`* = dom *voucherName` = dom *voucherStartDate`* = dom *voucherEndDate`*
= dom *voucherStatus`* = dom *voucherDiscount`*
∨
voucherId? ∈ *voucherId* ∧ *rep*! = *voucherExist* ∧ *voucherId* = dom *voucherName* =
dom *voucherStartDate* = dom *voucherEndDate*
= dom *voucherStatus* = dom *voucherDiscount* ∧ *voucherId`* = dom *voucherName`* =
dom *voucherStartDate`* = dom *voucherEndDate`*
= dom *voucherStatus`* = dom *voucherDiscount`*

RetrieveVoucherComplete ≙ (RetrieveVoucher ∧ Okay) ∨ RetrieveVoucherError

RetrieveVoucherComplete

$voucherId, voucherId` : VOUCHERID$

$voucherName, voucherName` : VOUCHERID \pfun VOUCHERNAME$

$voucherStartDate, voucherStarDate` : VOUCHERID \pfun VOUCHERSTARTDATE$

$voucherEndDate, voucherEndDate` : VOUCHERID \pfun VOUCHERENDDATE$

$voucherStatus, voucherStatus` = VOUCHERID \pfun VOUCHERSTATUS$

$voucherDiscount, voucherDiscount` : VOUCHERID \pfun VOUCHERDISCOUNT$

$voucherName? : \mathbb{P}\ VOUCHERNAME$

$voucherId! : \mathbb{P}\ VOUCHERID$

$voucherStartDate! : \mathbb{P}\ VOUCHERSTARTDATE$

$voucherEndDate! : \mathbb{P}\ VOUCHERENDDATE$

$voucherDiscount! : \mathbb{N}$

$voucherStatus! : VOUCHERSTATUS$

$rep! : RESPONSE$

$voucherName? \in \text{ran}\ voucherName$

$\land$

$voucherId! = \text{dom}\ (voucherName \rhd \{voucherName?\}) \land voucherName! = voucherName\ (\!|\{\ \text{dom}\ (voucherName \rhd \{voucherName?\})\ \}|\!) \land$

$voucherStartDate! = voucherStartDate\ (\!|\{\ \text{dom}\ (voucherName \rhd \{voucherName?\})\ \}|\!) \land$

$voucherEndDate! = voucherEndDate\ (\!|\{\ \text{dom}\ (voucherName \rhd \{voucherName?\})\ \}|\!) \land$

$voucherDiscount! = voucherDiscount\ (\!|\{\ \text{dom}\ (voucherName \rhd \{voucherName?\})\}|\!)$

$\land$

$voucherId = \text{dom}\ voucherName = \text{dom}\ voucherStartDate = \text{dom}\ voucherEndDate = \text{dom}\ voucherStatus = \text{dom}\ voucherDiscount$

$\land$

$voucherId` = \text{dom}\ voucherName` = \text{dom}\ voucherStartDate` = \text{dom}\ voucherEndDate` = \text{dom}\ voucherStatus` = \text{dom}\ voucherDiscount`$

$\lor$

$voucherName? \notin voucherName \land rep! = voucherNotExist \land \text{dom}\ voucherName = voucherId = \text{dom}\ voucherStartDate = \text{dom}\ voucherEndDate = \text{dom}\ voucherStatus = \text{dom}\ voucherDiscount \land voucherId` = \text{dom}\ voucherName` = \text{dom}\ voucherStartDate` = \text{dom}\ voucherEndDate` = \text{dom}\ voucherStatus` = \text{dom}\ voucherDiscount`$

$\land$

$(voucherId = voucherId` \land voucherName = voucherName` \land voucherStartDate = voucherStarDate` \land voucherEndDate = voucherEndDate` \land voucherStatus = voucherStatus` \land voucherDiscount = voucherDiscount`)$

UpdateVoucherComplete ≙ (UpdateVoucher ∧ Okay) ∨ UpdateVoucherError

UpdateVoucherComplete

$voucherId, voucherId` : VOUCHERID$
$voucherName, voucherName` : VOUCHERID \pfun VOUCHERNAME$
$voucherStartDate, voucherStarDate` : VOUCHERID \pfun VOUCHERSTARTDATE$
$voucherEndDate, voucherEndDate` : VOUCHERID \pfun VOUCHERENDDATE$
$voucherStatus, voucherStatus` = VOUCHERID \pfun VOUCHERSTATUS$
$voucherDiscount, voucherDiscount` : VOUCHERID \pfun VOUCHERDISCOUNT$

$voucherId? : \mathbb{P}\ VOUCHERID$
$voucherName? : \mathbb{P}\ VOUCHERNAME$
$voucherStartDate? : \mathbb{P}\ VOUCHERSTARTDATE$
$voucherEndDate? : \mathbb{P}\ VOUCHERENDDATE$
$voucherDiscount? : \mathbb{N}$
$voucherStatus? : VOUCHERSTATUS$
$rep! : RESPONSE$

$voucherId? \in voucherId$
$\land$
$(voucherName' = voucherName \oplus \{voucherId? \mapsto voucherName?\}$
$\lor$
$voucherStartDate' = voucherStartDate \oplus \{voucherId? \mapsto voucherStartDate?\}$
$\lor$
$voucherEndDate' = voucherEndDate \oplus \{voucherId? \mapsto voucherEndDate?\}$
$\lor$
$voucherStatus = voucherStatus \oplus \{voucherId? \mapsto voucherStatus?\}$
$\lor$
$voucherDiscount = voucherDiscount \oplus \{voucherId? \mapsto voucherDiscount?\})$
$\land$
$voucherId = \text{dom}\ voucherName = \text{dom}\ voucherStartDate = \text{dom}\ voucherEndDate$
$= \text{dom}\ voucherStatus = \text{dom}\ voucherDiscount$
$\land$
$voucherId` = \text{dom}\ voucherName` = \text{dom}\ voucherStartDate` = \text{dom}\ voucherEndDate`$
$= \text{dom}\ voucherStatus` = \text{dom}\ voucherDiscount`$
$\lor$
$voucherId? \notin voucherId \land rep! = voucherNotExist \land \text{dom}\ voucherName = voucherId =$
$\text{dom}\ voucherStartDate = \text{dom}\ voucherEndDate$
$= \text{dom}\ voucherStatus = \text{dom}\ voucherDiscount \land voucherId` = \text{dom}\ voucherName` =$
$\text{dom}\ voucherStartDate` = \text{dom}\ voucherEndDate`$
$= \text{dom}\ voucherStatus` = \text{dom}\ voucherDiscount`$

DeleteVoucherComplete ≙ (DeleteVoucher ∧ Okay) ∨ DeleteVoucherError

DeleteVoucherComplete

$$
\begin{array}{l}
voucherId, voucherId` : VOUCHERID \\
voucherName, voucherName` : VOUCHERID \pfun VOUCHERNAME \\
voucherStartDate, voucherStarDate` : VOUCHERID \pfun VOUCHERSTARTDATE \\
voucherEndDate, voucherEndDate` : VOUCHERID \pfun VOUCHERENDDATE \\
voucherStatus, voucherStatus` = VOUCHERID \pfun VOUCHERSTATUS \\
voucherDiscount, voucherDiscount` : VOUCHERID \pfun VOUCHERDISCOUNT \\
\\
voucherId? : \mathbb{P}\ VOUCHERID \\
rep! : RESPONSE \\
\hline
voucherId? \in voucherId \\
\land \\
voucherId' = voucherId \setminus voucherId? \land voucherName' = \{voucherId?\} \rhd voucherName \land \\
voucherStartDate' = \{voucherId?\} \rhd voucherStartDate \land voucherEndDate' = \\
\{voucherId?\} \rhd voucherEndDate \land \\
voucherDiscount' = \{voucherId?\} \rhd voucherDiscount \land voucherStatus' = \{voucherId?\} \rhd voucherStatus \\
\land \\
voucherId = \mathrm{dom}\ voucherName = \mathrm{dom}\ voucherStartDate = \mathrm{dom}\ voucherEndDate \\
= \mathrm{dom}\ voucherStatus = \mathrm{dom}\ voucherDiscount \\
\land \\
voucherId` = \mathrm{dom}\ voucherName` = \mathrm{dom}\ voucherStartDate` = \mathrm{dom}\ voucherEndDate` \\
= \mathrm{dom}\ voucherStatus` = \mathrm{dom}\ voucherDiscount` \\
\lor \\
voucherId? \notin voucherId \land rep! = voucherNotExist \land voucherId = \mathrm{dom}\ voucherName = \\
\mathrm{dom}\ voucherStartDate = \mathrm{dom}\ voucherEndDate \\
= \mathrm{dom}\ voucherStatus = \mathrm{dom}\ voucherDiscount \land voucherId` = \mathrm{dom}\ voucherName` = \\
\mathrm{dom}\ voucherStartDate` = \mathrm{dom}\ voucherEndDate` \\
= \mathrm{dom}\ voucherStatus` = \mathrm{dom}\ voucherDiscount`
\end{array}
$$

AddShopComplete ≙ (AddShop ∧ Okay) ∨ AddShopError

AddShopComplete

$shopId, shopId` : \mathbb{P}\ SHOPID$
$shopName, shopName` : SHOPID \pfun SHOPNAME$
$shopDesc, shopDesc` : SHOPID \pfun SHOPDESC$
$shopAdd, shopAdd` : SHOPID \pfun SHOPADDRESS$
$shopRating, shopRating` : SHOPID \pfun SHOPRATING$
$shopEmail, shopEmail` : SHOPID \pfun SHOPEMAIL$

$shopId? : \mathbb{P}\ SHOPID$
$shopName? : \mathbb{P}\ SHOPNAME$
$shopDesc? : \mathbb{P} SHOPDESC$
$shopAdd? : \mathbb{P}\ SHOPADDRESS$
$shopEmail? \ \mathbb{P}\ SHOPEMAIL$
$rep! : RESPONSE$

$shopId? \notin shopId$
$\land$
$shopId' = shopId \cup shopId? \land shopName' = shopName \cup \{shopId? \mapsto shopName?\} \land$
$shopDesc' = shopDesc \cup \{shopId? \mapsto shopDesc?\} \land shopAdd' = shopAdd \cup \{shopId? \mapsto shopAdd?\} \land$
$shopEmail' = shopEmail \cup \{shopId? \mapsto shopEmail?\}$
$\land$
$shopId = \text{dom}\ shopName = \text{dom}\ shopDesc = \text{dom}\ shopAdd = \text{dom}\ shopRating \land \#shopRating \leqslant maxRating$
$\land$
$shopId` = \text{dom}\ shopName` = \text{dom}\ shopDesc` = \text{dom}\ shopAdd` = \text{dom}\ shopRating` \land \#shopRating` \leqslant maxRating$

$(shopId? \in shopId \land rep! = shopExist \land shopId = \text{dom}\ shopName = \text{dom}\ shopDesc = \text{dom}\ shopAdd =$
$\text{dom}\ shopRating \land shopId` = \text{dom}\ shopName` = \text{dom}\ shopDesc` = \text{dom}\ shopAdd` = \text{dom}\ shopRating`)$
$\lor$
$(shopEmail? \in shopEmail \land rep! = emailExist \land shopId = \text{dom}\ shopName = \text{dom}\ shopDesc =$
$\text{dom}\ shopAdd = \text{dom}\ shopRating \land shopId` = \text{dom}\ shopName` = \text{dom}\ shopDesc` = \text{dom}\ shopAdd` =$
$\text{dom}\ shopRating`)$

RetrieveShopComplete ≙ (RetrieveShop ∧ Okay) ∨ RetrieveShopError

RetrieveShopComplete

$$
\begin{array}{l}
shopId, shopId` : \mathbb{P}\ SHOPID \\
shopName, shopName` : SHOPID \pfun SHOPNAME \\
shopDesc, shopDesc` : SHOPID \pfun SHOPDESC \\
shopAdd, shopAdd` : SHOPID \pfun SHOPADDRESS \\
shopRating, shopRating` : SHOPID \pfun SHOPRATING \\
shopEmail, shopEmail` : SHOPID \pfun SHOPEMAIL \\
\\
shopName? : SHOPNAME \\
shopId! : SHOPID \\
shopDesc! : SHOPDESC \\
shopAdd! : SHOPADDRESS \\
shopRating! : SHOPRATING \\
shopEmail! : SHOPEMAIL \\
rep! : RESPONSE \\
\\
\hline \\
shopName? \in \ran shopName \\
\land \\
shopId! = \dom (shopName \rhd \{shopName?\}) \land shopDesc! = shopDesc (\!|\{\dom (shopName \rhd \{shopName?\})\}|\!) \land \\
shopAdd! = shopAdd (\!|\{ \dom (shopName \rhd \{shopName?\})\}|\!) \land \\
shopRating! = shopAdd (\!|\{ \dom (shopName \rhd \{shopName?\})\}|\!) \land \\
shopeEmail! = shopAdd (\!|\{ \dom (shopName \rhd \{shopName?\})\}|\!) \\
\land \\
shopId = \dom shopName = \dom shopDesc = \dom shopAdd = \dom shopRating \land \#shopRating \leqslant maxRating \\
\land \\
shopId` = \dom shopName` = \dom shopDesc` = \dom shopAdd` = \dom shopRating` \land \#shopRating` \leqslant maxRating \\
\lor \\
(shopName? \notin shopName \land rep! = shopNotExist \land shopId = \dom shopName = \dom shopDesc = \\
\dom shopAdd = \dom shopRating \land shopId` = \dom shopName` = \dom shopDesc` = \dom shopAdd` = \\
\dom shopRating`) \\
\land \\
(shopId = shopId` \land shopName = shopName` \land shopDesc = shopDesc` \land \\
shopAdd = shopAdd` \land shopRating = shopRating` \land shopEmail = shopEmail`)
\end{array}
$$

UpdateShopComplete ≙ (UpdateShop ∧ Okay) ∨ UpdateShopError

UpdateShopComplete

$$
\begin{array}{l}
shopId, shopId` : \ \mathbb{P}\ SHOPID \\
shopName, shopName` : SHOPID \pfun SHOPNAME \\
shopDesc, shopDesc` : SHOPID \pfun SHOPDESC \\
shopAdd, shopAdd` : SHOPID \pfun SHOPADDRESS \\
shopRating, shopRating` : SHOPID \pfun SHOPRATING \\
shopEmail, shopEmail` : SHOPID \pfun SHOPEMAIL \\
\\
shopId? : \mathbb{P}\ SHOPID \\
shopName? : \mathbb{P}\ SHOPNAME \\
shopDesc? : \mathbb{P} SHOPDESC \\
shopAdd? : \mathbb{P}\ SHOPADDRESS \\
shopEmail? : \mathbb{P}\ SHOPEMAIL \\
shopRating? : \mathbb{N} \\
rep! : RESPONSE \\
\\
\hline \\
shopId? \in shopId \\
\land \\
(shopName' = shopName \oplus \{shopId? \mapsto shopName?\} \\
\lor \\
shopDesc' = shopDesc \oplus \{shopId? \mapsto shopDesc?\} \\
\lor \\
shopAdd' = shopAdd \oplus \{shopId? \mapsto shopAdd?\} \\
\lor \\
shopEmail' = shopEmail \oplus \{shopId? \mapsto shopEmail?\}) \\
\land \\
shopId = \mathrm{dom}\ shopName = \mathrm{dom}\ shopDesc = \mathrm{dom}\ shopAdd = \mathrm{dom}\ shopRating \land \#shopRating \leqslant maxRating \\
\land \\
shopId` = \mathrm{dom}\ shopName` = \mathrm{dom}\ shopDesc` = \mathrm{dom}\ shopAdd` = \mathrm{dom}\ shopRating` \land \#shopRating` \leqslant maxRating \\
\lor \\
(shopId? \notin shopId \land rep! = shopNotExist \land shopId = \mathrm{dom}\ shopName = \mathrm{dom}\ shopDesc = \\
\mathrm{dom}\ shopAdd = \mathrm{dom}\ shopRating \land shopId` = \mathrm{dom}\ shopName` = \mathrm{dom}\ shopDesc` = \mathrm{dom}\ shopAdd` = \\
\mathrm{dom}\ shopRating`) \\
\lor \\
(shopEmail? \in shopEmail \land rep! = shopEmailExist \land shopId = \mathrm{dom}\ shopName = \mathrm{dom}\ shopDesc = \\
\mathrm{dom}\ shopAdd = \mathrm{dom}\ shopRating \land shopId` = \mathrm{dom}\ shopName` = \mathrm{dom}\ shopDesc` = \mathrm{dom}\ shopAdd` = \\
\mathrm{dom}\ shopRating`)
\end{array}
$$

DeleteShopComplete ≙ (DeleteShop ∧ Okay) ∨ DeleteShopError

DeleteShopComplete

$$
\begin{array}{l}
shopId, shopId` : \mathbb{P}\ SHOPID \\
shopName, shopName` : SHOPID \pfun SHOPNAME \\
shopDesc, shopDesc` : SHOPID \pfun SHOPDESC \\
shopAdd, shopAdd` : SHOPID \pfun SHOPADDRESS \\
shopRating, shopRating` : SHOPID \pfun SHOPRATING \\
shopEmail, shopEmail` : SHOPID \pfun SHOPEMAIL \\
\\
shopId? : SHOPID \\
rep! : RESPONSE \\
\\
\hline
shopId? \in shopId \\
\land \\
shopId' = shopId \setminus shopId? \land shopName' = \{shopId?\} \ndres shopName \land \\
shopDesc' = \{shopId?\} \ndres shopDesc \land shopAdd' = \{shopId?\} \ndres shopAdd \land \\
shopRating' = \{shopId?\} \ndres shopRating \\
\land \\
shopId = \mathrm{dom}\ shopName = \mathrm{dom}\ shopDesc = \mathrm{dom}\ shopAdd = \mathrm{dom}\ shopRating \land \#shopRating \leqslant maxRating \\
\land \\
shopId` = \mathrm{dom}\ shopName` = \mathrm{dom}\ shopDesc` = \mathrm{dom}\ shopAdd` = \mathrm{dom}\ shopRating` \land \#shopRating` \leqslant maxRating \\
\lor \\
(shopId? \notin shopId \land rep! = shopNotExist \land shopId = \mathrm{dom}\ shopName = \mathrm{dom}\ shopDesc = \mathrm{dom}\ shopAdd = \\
\mathrm{dom}\ shopRating \land shopId` = \mathrm{dom}\ shopName` = \mathrm{dom}\ shopDesc` = \mathrm{dom}\ shopAdd` = \mathrm{dom}\ shopRating`) \\
\hline
\end{array}
$$

5.6 Conclusion

During these 12 weeks of lectures by Ms Azurawati and guidance by Ms Mazilina, I had learnt about the formal method technique to describe a software system.The Formal method help me to have more understanding and clear representation of a software system, especially how data will be input, how data will check errors and how it will output after processed. Besides, I had learnt about how the data should link together to represent their relationship inside the software system. It provides an alternative solution for me to represent my software system in the future. However, Formal method for software engineering is quite rarely applied in the market, but I believe that learning these techniques will allow me to prepare myself if facing any related project that is applying this kind of techniques.

In the end, I would like to appreciate the guidance from Ms Azurawati and Ms Mazilina on this assignment as formal methods have complicated data and function relationships with a lot of special symbols that represent different meanings. Besides, it also consists of standardized format to write all the declaration of data and schemas. However, I appreciate that I was given a chance to learn formal methods for software engineering to prepare myself for my future careers.

6.0 Cart Module

6.1 State Space Schema

Cart

cartId: $\mathbb{P}$ CARTID
cartProductId: (CARTID ⇸ PRODID)
cartUserId: (CARTID ⇸ USERID)
voucherId: (CARTPRODUCTID ⇸ VOUCHERID)
shopeeCoin: (CARTUSERID ⇸ SHOPEECOIN)
itemQty: (CARTPRODUCTID ⇸ $\mathbb{N}$)
subTotal: (CARTPRODUCTID ⇸ SUBTOTAL)
total: (CARTID ⇸ $\mathbb{N}$)

#itemQty ≤ maxAddCart
cartProductId = dom cartproductId = dom cartuserId = dom shopeeCoin = dom total
cartId = dom voucherId = dom itemQty = dom subTotal
cartUserId = dom shopeeCoin

6.2 Initial State Schema

InitCart

Cart
$cartId = \varnothing$
$cartProductId = \varnothing$
$cartUserId = \varnothing$
$voucherId = \varnothing$
$shopeeCoin = 0$
$itemQty = 0$
$subTotal = 0$
$totalPrice = 0$

6.3 Operation Schema

Operation 1: Create, Retrieve & Delete Cart

AddCart

$\Delta Cart$
$\Xi User$
$\Xi Product$
$prodID? : PRODID$
$user? : USERID$
$qty? : \mathbb{N}$

$(user? \in userId \land userStatus(\!|\{user?\}|\!) = loggedIn)$
$prodID? \notin cartProductId$
$\#prodID? \leqslant maxAddCart$
$cartProductId' = cartProductId \cup (cartId \mapsto dom\ prodName?)$
$cartId' = cartId \cup cartProductId$
$itemQty' = itemQty \cup \{product? \pfun qty?\}$
$subTotal' = subTotal \oplus (subTotal \mapsto (prodPrice\{prodID?\} \times itemQty))$
$total' = total \oplus (cartId \mapsto \forall subTotal\}$
$prodId' = prodId$
$prodName' = prodName$
$prodDetail' = prodDetail$
$prodPrice' = prodPrice$
$prodStock' = prodStock$

DeleteCart

$\Delta Cart$
$\Xi User$
$\Xi Product$
cartProdID?: *CARTPRODUCTID*
user?: *USERID*

$(user? \in userId \land userStatus(\{user?\}) = loggedIn)$
$cartprodID? \subseteq cartId$
$cartId' = \{cartProdID?\} \lhd cartId$
$itemQty' = \{cartProdID?\} \lhd itemQty$
$subTotal' = \{cartProdID?\} \lhd subTotal$
$total' = total \oplus \{cartId \mapsto \forall subTotal\}$
$prodId' = prodId$
$prodName' = prodName$
$prodDetail' = prodDetail$
$prodPrice' = prodPrice$
$prodStock' = prodStock$
$user' = user$

EditCart

$\Delta Cart$
$\Xi Product$
$\Xi USER$
cartProdID?: *CARTPRODUCTID*
user?: *USERID*
qty? : $\mathbb{N}$

$(user? \in userId \land userStatus(\{user?\}) = loggedIn)$
$cartprodID? \subseteq cartId$
$\#qty? \geqslant 1$
$itemQty' = itemQty \oplus ((itemQty \cup \{qty?\}) \vee (itemQty \setminus \{qty?\}))$
$subTotal' = productPrice * itemQty$
$subTotal = subTotal \oplus (subTotal \mapsto (prodPrice\{prodID?\} \times itemQty))$
$total = total \oplus (cartId \mapsto \forall subTotal\}$
$prodId' = prodId$
$prodName' = prodName$
$prodDetail' = prodDetail$
$prodPrice' = prodPrice$
$prodStock' = prodStock$
$cartId' = cartId$

Operation 2: Apply Voucher

ApplyVoucher

$\Delta Cart$
$\Delta Voucher$
$\Xi User$
cartProdID?: *CARTPRODUCTID*
user?: *USERID*
voucher?: *VOUCHERID*

$(user? \in userId \land userStatus(\{user?\}) = loggedIn)$
$cartprodID? \subseteq cartId$
$voucher' = \{cartprodID?\} \mapsto \{voucher?\}$
$\{voucher?\} \mapsto voucherStatus == valid$
$subTotal = subTotal \oplus (subTotal \mapsto subTotal(cartProdID) \setminus voucherDiscount)$
$total' = total \oplus (cartId \mapsto \forall subTotal\}$
$\{voucher?\} \mapsto voucherStatus = used$
$voucher' = voucher \oplus \{voucher?\}$
$itemQty' = itemQty$
$cartId' = cartId$

Operation 3: Redeem Shopee Coin

ReedemShopeeCoin

$\Delta Cart$

$\Xi User$

cart?: *CARTID*

user?: *USERID*

$(user? \in userId \land userStatus(\!|\{user?\}|\!) = loggedIn)$

$cartprodID? \subseteq cartId$

$shopeeCoin = \{user?\} \mapsto shopeeCoin$

$\#shopeeCoin \geqslant 10$

$total' = total \oplus (cart? \mapsto (total \setminus (shopeeCoin \div 100))$

$shopeeCoin' = shopeeCoin \oplus (\{user?\} \mapsto 0)$

$userId' = userId$

$itemQty' = itemQty$

$cartId' = cartId$

6.4 Error Scenarios

Error Scenario Table

Schema Nme	Success Pre-Condition	Failure Pre-Condition	Remark
AddCart	prodId? ∉ cartProductId #qty? ⩽ maxAddCart *(user? ∈ userId ∧ userStatus(|{user?}|) = loggedIn)*	prodId? ∈ cartProductId #qty? > maxAddCart prodStock == 0 (*user? ∈ userId ∧ userStatus(|{user?}|) = loggedOut)*	productExist invalidQuantity outOfStock userNotLoggedIn
DeleteCart	cartProdID? ⊆ cartId *(user? ∈ userId ∧ userStatus(|{user?}|) = loggedIn)*	cartProdID? ⊄ cartId (*user? ∈ userId ∧ userStatus(|{user?}|) = loggedOut)*	cartProductNotExist userNotLoggedIn
EditCart	cartProdID? ⊆ cartId #qty? ⩾ 1 *(user? ∈ userId ∧ userStatus(|{user?}|) = loggedIn)*	cartProdID? ⊄ cartId #qty? < 1 (*user? ∈ userId ∧ userStatus(|{user?}|) = loggedOut)*	cartProductNotExist invalidQuantity userNotLoggedIn
ApplyVoucher	cartProdID? ⊆ cartId {*voucher?*} ↦ *voucherStatus == valid* *(user? ∈ userId ∧ userStatus(|{user?}|) = loggedIn)*	cartProdID? ⊄ cartId {*voucher?*} ↦ *voucherStatus* == in*valid* (*user? ∈ userId ∧ userStatus(|{user?}|) = loggedOut)*	cartProductNotExist voucherStatusInvalid userNotLoggedIn
ReedemShopeeCoin	cartProdID? ⊆ cartId *#shopeeCoin* ⩾ 10 *(user? ∈ userId ∧ userStatus(|{user?}|) = loggedIn)*	cartProdID? ⊄ cartId *#shopeeCoin* < 10 (*user? ∈ userId ∧ userStatus(|{user?}|) = loggedOut)*	cartProductNotExist insufficientShopeeCoin userNotLoggedIn

Error Scenario Free Type

RESPONSECART ::= success | productExist | invalidQuantity | cartProductNotExist | outOfStock | voucherStatusInvalid | insufficientShopeeCoin | userNotLoggedIn

Okay

$rep! : RESPONSECART$

$rep! = success$

Error Scenario

AddCartError

$\Xi Cart$
$\Xi Product$
$\Xi User$
$user?: USERID$
$prodID?: PRODID$
$qty?: \mathbb{N}$
$rep!: RESPONSECART$

$(prodId? \in cartProductId \land rep! = productExist)$
$\lor$
$(\#qty? > maxAddCart \land rep! = invalidQuantity)$
$\lor$
$(prodStock == 0 \land rep! = outOfStock)$
$\lor$
$(user? \in userId \land userStatus(\!|\{user?\}|\!) = loggedOut \land rep! = userNotLoggedIn)$

DeleteCartError

$\Xi Cart$
$\Xi Product$
$\Xi User$
$user?: USERID$
$cartProdID?: CARTPRODUCTID$
$rep!: RESPONSECART$

$cartProdID? \notin cartId \land rep! = cartProductNotExist$
$\lor$
$(user? \in userId \land userStatus(\!|\{user?\}|\!) = loggedOut \land rep! = userNotLoggedIn)$

EditCartError

$\Xi Cart$
$\Xi Product$
$\Xi User$
user?: *USERID*
cartProdID?: *CARTPRODUCTID*
qty?: $\mathbb{N}$
rep!: *RESPONSECART*

(cartProdID? $\notin$ cartId $\land$ *rep*! = *cartProductNotExist*)
$\lor$
(#*qty*? < 1 $\land$ *rep*! = *invalidQuantity*)
$\lor$
(*user?* $\in$ *userId* $\land$ *userStatus*(|{*user?*}|) = *loggedOut* $\land$ *rep*! = *userNotLoggedIn)*

ApplyVoucherError

$\Xi Voucher$
$\Xi Cart$
$\Xi User$
user?: *USERID*
cartProdID?: *CARTPRODID*
voucher?: *VOUCHERID*
rep!: *RESPONSECART*

(cartProdID? $\notin$ cartId $\land$ *rep*! = *cartProductNotExist*)
$\lor$
({*voucher?*} $\mapsto$ *voucherStatus* == *invalid* $\land$ *rep*! = *voucherInvalid*)
$\lor$
(*user?* $\in$ *userId* $\land$ *userStatus*(|{*user?*}|) = *loggedOut* $\land$ *rep*! = *userNotLoggedIn)*

RedeemShopeeCoinError

$\Xi Cart$
$\Xi User$
cart? : *CARTID*
user?: *USERID*
rep!: *RESPONSECART*

(*user?* $\in$ *userId* $\land$ *userStatus*(|{*user?*}|) = *loggedOut* $\land$ *rep*! = *userNotLoggedIn)*
$\lor$
(cartProdID? $\notin$ cartId $\land$ *rep*! = *cartProductNotExist*)
$\lor$
(#*shopeeCoin* < 10 $\land$ *rep*! = *insufficientShopeeCoin*)]

6.5 Complete Schema

AddCartComplete ≙ (AddCart ∧ Okay) ∨ AddCartError

AddCartComplete

$cartId, cartId' : \mathbb{P}\ CARTID$

$cartProductId, cartProductId' : \mathbb{P}\ (CARTID \pfun PRODID)$

$cartUserId, cartUserId' : \mathbb{P}\ (CARTID \pfun USERID)$

$voucherID, voucherID' : \mathbb{P}\ (CARTPRODUCTID \pfun VOUCHERID)$

$shopeeCoin, shopeeCoin' : (CARTUSERID \pfun SHOPEECOIN)$

$itemQty, itemQty' : (CARTPRODUCTID \pfun \mathbb{N})$

$subTotal, subTotal' : (CARTPRODUCTID \pfun SUBTOTAL)$

$total, total' : (CARTID \pfun \mathbb{N})$

$userId, userId' : \mathbb{P}\ USERID$

$username, userName' : \mathbb{P}\ (USERID \times USERNAME)$

$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$

$userPhone , userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$

$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$

$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$

$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$prodId, prodId' : \mathbb{P}\ PRODUCTID$

$prodName, prodName' : \mathbb{P}\ (PRODUCTID \times PRODUCTNAME)$

$prodDetail, prodDetail' : \mathbb{P}\ (PRODUCTID \times PRODUCTDETAIL)$

$prodStock, prodStock' : \mathbb{P}\ (PRODUCTID \times PRODUCTSTOCK)$

$prodPrice, prodPrice' : \mathbb{P}\ (PRODUCTID \times PRODUCTPRICE)$

$prodCategotry, prodCategotry' : \mathbb{P}(PRODUCTID \times PRODUCT\ CATEGORY)$

$user? : USERID$

$prodID? : PRODID$

$qty? : \mathbb{N}$

$rep! : RESPONSECART$

$(user? \in userId \land userStatus(\!|\{user?\}|\!) = loggedIn) \land prodID? \notin cartProductId \land$

$\#prodID? \leqslant maxAddCart \land cartProductId = cartProductId \cup (cartId \mapsto dom\ prodName?) \land$

$cartId = cartId \cup cartProductId \land itemQty = itemQty \cup \{product? \pfun qty?\} \land$

$subTotal = subTotal \oplus (subTotal \mapsto (prodPrice\{prodID?\} \times itemQty)) \land$

$total = total \oplus (cartId \mapsto \forall subTotal\} \land prodId' = prodId \land$

$prodName' = prodName \land\ prodDetail' = prodDetail \land prodPrice' = prodPrice \land$

$$
\begin{aligned}
&prodStock' = prodStock \land cartId' = cartId \land rep! = success \land \\
&\#itemQty \leqslant maxAddCart \land \#itemQty' \leqslant maxAddCart \land \\
&dom\ cartProductId = dom\ cartUserId = dom\ shopeeCoin = dom\ total = cartId \land \\
&dom\ cartProductId' = dom\ cartUserId' = dom\ shopeeCoin' = dom\ total = cartId' \land \\
&dom\ voucherID = dom\ itemQty = dom\ subTotal = cartProductId \land \\
&dom\ voucherID' = dom\ itemQty' = dom\ subTotal' = cartProductId' \land \\
&\text{dom } prodName = \text{dom } prodDetail = \text{dom } prodStock = \text{dom } prodPrice = \text{dom } prodCategotry = prodId \land \\
&\text{dom } prodName' = \text{dom } prodDetail' = \text{dom } prodStock' = \text{dom } prodPrice' = \text{dom } prodCategotry' = prodId' \land \\
&dom\ userName = dom\ userEmail = dom\ userGender = dom\ userPhone = dom\ password = \\
&dom\ userHistory = dom\ userStatus = userId \land \\
&dom\ userName' = dom\ userEmail' = dom\ userGender' = dom\ userPhone' = dom\ password' = \\
&dom\ userHistory' = dom\ userStatus' = userId') \\
&\lor \\
&(\\
&(((prodId? \in cartProductId \land rep! = productExist \land \\
&dom\ voucherID = dom\ itemQty = dom\ subTotal = cartProductId \land \\
&dom\ voucherID' = dom\ itemQty' = dom\ subTotal' = cartProductId' \land \\
&\text{dom } prodName = \text{dom } prodDetail = \text{dom } prodStock = \text{dom } prodPrice = \text{dom } prodCategotry = prodId \land \\
&\text{dom } prodName' = \text{dom } prodDetail' = \text{dom } prodStock' = \text{dom } prodPrice' = \text{dom } prodCategotry' = prodId') \\
&\lor \\
&(\#qty? > maxAddCart \land rep! = invalidQuantity \land \\
&\#itemQty \leqslant maxAddCart \land \#itemQty' \leqslant maxAddCart) \\
&\lor \\
&(prodStock == 0 \land rep! = outOfStock \land \\
&\#PRODUCTSTOCK \leqslant maxQuantity \land \#PRODUCTSTOCK' \leqslant maxQuantity) \\
&\lor \\
&(user? \in userId \land userStatus(\!|\{user?\}|\!) = loggedOut \land rep! = userNotLoggedIn) \land \\
&dom\ userName = dom\ userEmail = dom\ userGender = dom\ userPhone = dom\ password = \\
&dom\ userHistory = dom\ userStatus = userId \land \\
&dom\ userName' = dom\ userEmail' = dom\ userGender' = dom\ userPhone' = dom\ password' = \\
&dom\ userHistory' = dom\ userStatus' = userId')) \\
&\land \\
&(prodId' = prodId \land qty' = qty \land prodStock' = prodStock \land user' = user) \\
&)
\end{aligned}
$$

DeleteCartComplete = (DeleteCart ∧ Okay) ∨ DeleteCartError

SchemaName

$cartId, cartId' : \mathbb{P}\ CARTID$
$cartProductId, cartProductId' : \mathbb{P}\ (CARTID \pfun PRODID)$
$cartUserId, cartUserId' : \mathbb{P}\ (CARTID \pfun USERID)$
$voucherID, voucherID' : \mathbb{P}\ (CARTPRODUCTID \pfun VOUCHERID)$
$shopeeCoin, shopeeCoin' : (CARTUSERID \pfun SHOPEECOIN)$
$itemQty, itemQty' : (CARTPRODUCTID \pfun \mathbb{N})$
$subTotal, subTotal' : (CARTPRODUCTID \pfun SUBTOTAL)$
$total, total' : (CARTID \pfun \mathbb{N})$

$userId, userId' : \mathbb{P}\ USERID$
$username, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone , userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$prodId, prodId' : \mathbb{P}\ PRODUCTID$
$prodName, prodName' : \mathbb{P}\ (PRODUCTID \times PRODUCTNAME)$
$prodDetail, prodDetail' : \mathbb{P}\ (PRODUCTID \times PRODUCTDETAIL)$
$prodStock, prodStock' : \mathbb{P}\ (PRODUCTID \times PRODUCTSTOCK)$
$prodPrice, prodPrice' : \mathbb{P}\ (PRODUCTID \times PRODUCTPRICE)$
$prodCategotry, prodCategotry' : \mathbb{P}(PRODUCTID \times PRODUCT\ CATEGORY)$

$cartProdID? : CARTPRODUCTID$
$user? : USERID$
$rep! : RESPONSECART$

$(user? \in userId \land userStatus(\!|\{user?\}|\!) = loggedIn) \land cartprodID? \subseteq cartId \land$
$cartId' = \{cartProdID?\} \ndres cartId \land itemQty' = \{cartProdID?\} \ndres itemQty \land$
$subTotal' = \{cartProdID?\} \ndres subtotal \land total' = total \oplus \{cartId \mapsto \forall subTotal\} \land prodId' = prodId$
$prodName' = prodName \land prodDetail' = prodDetail \land prodPrice' = prodPrice \land$
$prodStock' = prodStock \land cartId' = cartId \land rep! = success \land$
$\#itemQty \leqslant maxAddCart \land \#itemQty' \leqslant maxAddCart \land$
$dom\ cartProductId = dom\ cartUserId = dom\ shopeeCoin = dom\ total = cartId \land$
$dom\ cartProductId' = dom\ cartUserId' = dom\ shopeeCoin' = dom\ total = cartId' \land$
$dom\ voucherID = dom\ itemQty = dom\ subTotal = cartProductId \land$
$dom\ voucherID' = dom\ itemQty' = dom\ subTotal' = cartProductId' \land$
$dom\ prodName = dom\ prodDetail = dom\ prodStock = dom\ prodPrice = dom\ prodCategotry = prodId \land$
$dom\ prodName' = dom\ prodDetail' = dom\ prodStock' = dom\ prodPrice' = dom\ prodCategotry' = prodId' \land$

$$
\begin{aligned}
&dom\ userName = dom\ userEmail = dom\ userGender = dom\ userPhone\ = dom\ password = \\
&dom\ userHistory = dom\ userStatus = userId \land \\
&dom\ userName' = dom\ userEmail' = dom\ userGender' = dom\ userPhone'\ = dom\ password' = \\
&dom\ userHistory' = dom\ userStatus' = userId') \\
&\lor \\
&(((cartProdID? \notin cartId \land rep! = cartProductNotExist \land \\
&dom\ cartProductId = dom\ cartUserId = dom\ shopeeCoin = dom\ total = cartId \land \\
&dom\ cartProductId' = dom\ cartUserId' = dom\ shopeeCoin' = dom\ total = cartId') \\
&\lor \\
&(user? \in userId \land userStatus(\!|\{user?\}|\!) = loggedOut \land rep! = userNotLoggedIn \land \\
&dom\ userName = dom\ userEmail = dom\ userGender = dom\ userPhone\ = dom\ password = \\
&dom\ userHistory = dom\ userStatus = userId \land \\
&dom\ userName' = dom\ userEmail' = dom\ userGender' = dom\ userPhone'\ = dom\ password' = \\
&dom\ userHistory' = dom\ userStatus' = userId')) \\
&\land \\
&(cartProdId' = cartProdId \land user' = user))
\end{aligned}
$$

EditCartComplete = (EditCart ∧ Okay) ∨ EditCartError

EditCartComplete

$cartId, cartId' : \mathbb{P}\ CARTID$

$cartProductId, cartProductId' : \mathbb{P}\ (CARTID \pfun PRODID)$

$cartUserId, cartUserId' : \mathbb{P}\ (CARTID \pfun USERID)$

$voucherId, voucherId' : \mathbb{P}\ (CARTPRODUCTID \pfun VOUCHERID)$

$shopeeCoin, shopeeCoin' : (CARTUSERID \pfun SHOPEECOIN)$

$itemQty, itemQty' : (CARTPRODUCTID \pfun \mathbb{N})$

$subTotal, subTotal' : (CARTPRODUCTID \pfun SUBTOTAL)$

$total, total' : (CARTID \pfun \mathbb{N})$

$userId, userId' : \mathbb{P}\ USERID$
$username, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone , userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$prodId, prodId' : \mathbb{P}\ PRODUCTID$
$prodName, prodName' : \mathbb{P}\ (PRODUCTID \times PRODUCTNAME)$
$prodDetail, prodDetail' : \mathbb{P}\ (PRODUCTID \times PRODUCTDETAIL)$
$prodStock, prodStock' : \mathbb{P}\ (PRODUCTID \times PRODUCTSTOCK)$
$prodPrice, prodPrice' : \mathbb{P}\ (PRODUCTID \times PRODUCTPRICE)$
$prodCategotry, prodCategotry' : \mathbb{P}(PRODUCTID \times PRODUCT\ CATEGORY)$

$cartProdID? : CARTPRODUCTID$
$user? : USERID$
$qty? : \mathbb{N}$
$rep! : RESPONSECART$

$((user? \in userId \land userStatus(\!|\{user?\}|\!) = loggedIn) \land cartprodID? \subseteq cartId \land \#qty? \geqslant 1 \land$
$itemQty' = itemQty \oplus ((itemQty \cup \{qty?\}) \lor (itemQty \setminus \{qty?\})) \land subTotal' = productPrice * itemQty \land$
$subTotal = subTotal \oplus (subTotal \mapsto (prodPrice\{prodID?\} \times itemQty)) \land$
$total = total \oplus (cartId \mapsto \forall subTotal\} \land prodId' = prodId \land prodName' = prodName \land$
$prodDetail' = prodDetail \land prodPrice' = prodPrice \land prodStock' = prodStock \land cartId' = cartId \land$
$rep! = success \land \#itemQty \leqslant maxAddCart \land \#itemQty' \leqslant maxAddCart \land$

$$
\begin{aligned}
&\mathit{dom\ cartProductId} = \mathit{dom\ cartUserId} = \mathit{dom\ shopeeCoin} = \mathit{dom\ total} = \mathit{cartId} \land \\
&\mathit{dom\ cartProductId'} = \mathit{dom\ cartUserId'} = \mathit{dom\ shopeeCoin'} = \mathit{dom\ total} = \mathit{cartId'} \land \\
&\mathit{dom\ voucherID} = \mathit{dom\ itemQty} = \mathit{dom\ subTotal} = \mathit{cartProductId} \land \\
&\mathit{dom\ voucherID'} = \mathit{dom\ itemQty'} = \mathit{dom\ subTotal'} = \mathit{cartProductId'} \land \\
&\mathit{dom\ userName} = \mathit{dom\ userEmail} = \mathit{dom\ userGender} = \mathit{dom\ userPhone} = \mathit{dom\ password} = \\
&\mathit{dom\ userHistory} = \mathit{dom\ userStatus} = \mathit{userId} \land \\
&\mathit{dom\ userName'} = \mathit{dom\ userEmail'} = \mathit{dom\ userGender'} = \mathit{dom\ userPhone'} = \mathit{dom\ password'} = \\
&\mathit{dom\ userHistory'} = \mathit{dom\ userStatus'} = \mathit{userId'} \land \\
&\mathit{dom\ userName} = \mathit{dom\ userEmail} = \mathit{dom\ userGender} = \mathit{dom\ userPhone} = \mathit{dom\ password} = \\
&\mathit{dom\ userHistory} = \mathit{dom\ userStatus} = \mathit{userId} \land \\
&\mathit{dom\ userName'} = \mathit{dom\ userEmail'} = \mathit{dom\ userGender'} = \mathit{dom\ userPhone'} = \mathit{dom\ password'} = \\
&\mathit{dom\ userHistory'} = \mathit{dom\ userStatus'} = \mathit{userId'}) \\
\lor\ &(((\mathit{cartProdID?} \notin \mathit{cartId} \land \mathit{rep!} = \mathit{cartProductNotExist} \land \\
&\mathit{dom\ cartProductId} = \mathit{dom\ cartUserId} = \mathit{dom\ shopeeCoin} = \mathit{dom\ total} = \mathit{cartId} \land \\
&\mathit{dom\ cartProductId'} = \mathit{dom\ cartUserId'} = \mathit{dom\ shopeeCoin'} = \mathit{dom\ total} = \mathit{cartId'} \land \\
&\mathit{dom\ voucherID} = \mathit{dom\ itemQty} = \mathit{dom\ subTotal} = \mathit{cartProductId} \land \\
&\mathit{dom\ voucherID'} = \mathit{dom\ itemQty'} = \mathit{dom\ subTotal'} = \mathit{cartProductId'}) \\
\lor\ &(\#\mathit{qty?} < 1 \land \mathit{rep!} = \mathit{invalidQuantity} \land \#\mathit{itemQty} \leqslant \mathit{maxAddCart} \land \#\mathit{itemQty'} \leqslant \mathit{maxAddCart}) \\
\lor\ &(\mathit{user?} \in \mathit{userId} \land \mathit{userStatus}(\!|\{\mathit{user?}\}|\!) = \mathit{loggedOut} \land \mathit{rep!} = \mathit{userNotLoggedIn} \land \\
&\mathit{dom\ userName} = \mathit{dom\ userEmail} = \mathit{dom\ userGender} = \mathit{dom\ userPhone} = \mathit{dom\ password} = \\
&\mathit{dom\ userHistory} = \mathit{dom\ userStatus} = \mathit{userId} \land \\
&\mathit{dom\ userName'} = \mathit{dom\ userEmail'} = \mathit{dom\ userGender'} = \mathit{dom\ userPhone'} = \mathit{dom\ password'} = \\
&\mathit{dom\ userHistory'} = \mathit{dom\ userStatus'} = \mathit{userId'}) \\
\land\ &\mathit{cartProdID'} = \mathit{cartProdID} \land \mathit{qty'} = \mathit{qty} \land \mathit{user'} = \mathit{user})
\end{aligned}
$$

ApplyVoucherComplete = (ApplyVoucher ∧ Okay) ∨ ApplyVoucherError

ApplyVoucherComplete

$cartId, cartId' : \mathbb{P}\ CARTID$
$cartProductId, cartProductId' : \mathbb{P}\ (CARTID \pfun PRODID)$
$cartUserId, cartUserId' : \mathbb{P}\ (CARTID \pfun USERID)$
$voucherID, voucherID' : \mathbb{P}\ (CARTPRODUCTID \pfun VOUCHERID)$
$shopeeCoin, shopeeCoin' : (CARTUSERID \pfun SHOPEECOIN)$
$itemQty, itemQty' : (CARTPRODUCTID \pfun \mathbb{N})$
$subTotal, subTotal' : (CARTPRODUCTID \pfun SUBTOTAL)$
$total, total' : (CARTID \pfun \mathbb{N})$

$userId, userId' : \mathbb{P}\ USERID$
$username, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone , userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$voucherId, voucherId' : \mathbb{P}\ VOUCHERID$
$voucherName, voucherName' : VOUCHERID \mapsto VOUCHERNAME$
$voucherStartDate, voucherStartDate' : VOUCHERID \mapsto VOUCHERSTARTDATE$
$voucherEndDate, voucherEndDate' : VOUCHERID \mapsto VOUCHERENDDATE$
$voucherStatus, voucherStatus' : VOUCHERID \mapsto VOUCHERSTATUS$
$voucherDiscount, voucherDiscount' : VOUCHERID \mapsto VOUCHERDISCOUNT$

$cartProdID? : CARTPRODUCTID$
$user? : USERID$
$voucher? : VOUCHERID$
$rep! : RESPONSECART$

$(user? \in userId \land userStatus(\!|\{user?\}|\!) = loggedIn) \land cartprodID? \subseteq cartId \land$
$voucher' = \{cartprodID?\} \mapsto \{voucher?\} \land \{voucher?\} \mapsto voucherStatus == valid \land$
$subTotal = subTotal \oplus (subTotal \mapsto subTotal(cartProdID) \setminus voucherDiscount) \land$
$total' = total \oplus (cartId \mapsto \forall subTotal\} \land \{voucher?\} \mapsto voucherStatus = used \land$
$voucher' = voucher \oplus \{voucher?\} \land itemQty' = itemQty \land cartId' = cartId \land rep! = success \land$
$\#itemQty \leqslant maxAddCart \land \#itemQty' \leqslant maxAddCart \land$
$dom\ cartProductId = dom\ cartUserId = dom\ shopeeCoin = dom\ total = cartId \land$
$dom\ cartProductId' = dom\ cartUserId' = dom\ shopeeCoin' = dom\ total = cartId' \land$
$dom\ voucherID = dom\ itemQty = dom\ subTotal = cartProductId \land$
$dom\ voucherID' = dom\ itemQty' = dom\ subTotal' = cartProductId' \land$
$dom\ voucherName = dom\ voucherStartDate = dom\ voucherEndDate = dom\ voucherStatus =$

$$
\begin{aligned}
&dom\ voucherDiscount = voucherId \land \\
&\text{dom } voucherName' = \text{dom } voucherStartDate' = \text{dom } voucherEndDate' = \text{dom } voucherStatus' = \\
&dom\ voucherDiscount' = voucherId' \land \\
&dom\ userName = dom\ userEmail = dom\ userGender = dom\ userPhone\ = dom\ password = \\
&dom\ userHistory = dom\ userStatus = userId \land \\
&dom\ userName' = dom\ userEmail' = dom\ userGender' = dom\ userPhone'\ = dom\ password' = \\
&dom\ userHistory' = dom\ userStatus' = userId')) \\
\lor\ &(((\text{cartProdID?} \notin \text{cartId} \land rep! = cartProductNotExist \land \\
&dom\ cartProductId = dom\ cartUserId = dom\ shopeeCoin = dom\ total = cartId \land \\
&dom\ cartProductId' = dom\ cartUserId' = dom\ shopeeCoin' = dom\ total = cartId' \land \\
&dom\ voucherID = dom\ itemQty = dom\ subTotal = cartProductId \land \\
&dom\ voucherID' = dom\ itemQty' = dom\ subTotal' = cartProductId') \\
\lor\ &(\{voucher?\} \mapsto voucherStatus == invalid \land rep! = voucherInvalid \land \\
&\text{dom } voucherName = \text{dom } voucherStartDate = \text{dom } voucherEndDate = \text{dom } voucherStatus = \\
&dom\ voucherDiscount = voucherId \land \\
&\text{dom } voucherName' = \text{dom } voucherStartDate' = \text{dom } voucherEndDate' = \text{dom } voucherStatus' = \\
&dom\ voucherDiscount' = voucherId'))) \\
\lor\ &(user? \in userId \land userStatus(\!|\{user?\}|\!) = loggedOut \land rep! = userNotLoggedIn \land \\
&dom\ userName = dom\ userEmail = dom\ userGender = dom\ userPhone\ = dom\ password = \\
&dom\ userHistory = dom\ userStatus = userId \land \\
&dom\ userName' = dom\ userEmail' = dom\ userGender' = dom\ userPhone'\ = dom\ password' = \\
&dom\ userHistory' = dom\ userStatus' = userId')) \\
\land\ &(cartProdID' = cartProdID \land voucher' = voucher \land user' = user))
\end{aligned}
$$

RedeemShopeeCoinComplete = (RedeemShopeeCoin ∧ Okay) ∨ RedeemShopeeCoinError

RedeemShopeeCoinComplete

$cartId, cartId' : \mathbb{P}\ CARTID$
$cartProductId,\ cartProductId' : \mathbb{P}\ (CARTID \pfun PRODID)$
$cartUserId,\ cartUserId' : \mathbb{P}\ (CARTID \pfun USERID)$
$voucherID,\ voucherID' : \mathbb{P}\ (CARTPRODUCTID \pfun VOUCHERID)$
$shopeeCoin,\ shopeeCoin' : (CARTUSERID \pfun SHOPEECOIN)$
$itemQty,\ itemQty' : (CARTPRODUCTID \pfun \mathbb{N})$
$subTotal,\ subTotal' : (CARTPRODUCTID \pfun SUBTOTAL)$
$total,\ total' : (CARTID \pfun \mathbb{N})$

$userId,\ userId' : \mathbb{P}\ USERID$
$username,\ userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender,\ userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone\ ,\ userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword,\ userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory,\ userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus,\ userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$cart? : CARTID$
$user? : USERID$
$rep! : RESPONSECART$

$(user? \in userId \land userStatus(\!|\{user?\}|\!) = loggedIn) \land cartprodID? \subseteq cartId \land$
$shopeeCoin = \{user?\} \mapsto shopeeCoin \land \#shopeeCoin \geqslant 10$
$total' = total \oplus (cart? \mapsto (total \setminus (shopeeCoin \div 100)) \land shopeeCoin' = shopeeCoin \oplus (\{user?\} \mapsto 0) \land$
$userId' = userId \land itemQty' = itemQty \land cartId' = cartId \land rep! = success \land$
$\#itemQty \leqslant maxAddCart \land \#itemQty' \leqslant maxAddCart \land$
$dom\ cartProductId = dom\ cartUserId = dom\ shopeeCoin = dom\ total = cartId \land$
$dom\ cartProductId' = dom\ cartUserId' = dom\ shopeeCoin' = dom\ total = cartId' \land$
$dom\ voucherID = dom\ itemQty = dom\ subTotal = cartProductId \land$
$dom\ voucherID' = dom\ itemQty' = dom\ subTotal' = cartProductId'$
$dom\ shopeeCoin = cartUserId \land dom\ shopeeCoin' = cartUserId' \land$
$dom\ userName = dom\ userEmail = dom\ userGender = dom\ userPhone\ = dom\ password =$
$dom\ userHistory = dom\ userStatus = userId \land$
$dom\ userName' = dom\ userEmail' = dom\ userGender' = dom\ userPhone'\ = dom\ password' =$
$dom\ userHistory' = dom\ userStatus' = userId' \land$
$dom\ userName = dom\ userEmail = dom\ userGender = dom\ userPhone\ = dom\ password =$
$dom\ userHistory = dom\ userStatus = userId \land$
$dom\ userName' = dom\ userEmail' = dom\ userGender' = dom\ userPhone'\ = dom\ password' =$

$$
\begin{array}{l}
dom\ userHistory' = dom\ userStatus' = userId') \\
\vee \\
(\\
((user? \in userId \curlywedge userStatus(\!|\{user?\}|\!) = loggedOut \wedge rep! = userNotLoggedIn \wedge \\
dom\ userName = dom\ userEmail = dom\ userGender = dom\ userPhone\ \ = dom\ password = \\
dom\ userHistory = dom\ userStatus = userId \wedge \\
dom\ userName' = dom\ userEmail' = dom\ userGender' = dom\ userPhone'\ \ = dom\ password' = \\
dom\ userHistory' = dom\ userStatus' = userId') \\
\vee \\
(cartProdID? \notin cartId \wedge rep! = cartProductNotExist \wedge \\
dom\ cartProductId = dom\ cartUserId = dom\ shopeeCoin = dom\ total = cartId \wedge \\
dom\ cartProductId' = dom\ cartUserId' = dom\ shopeeCoin' = dom\ total = cartId' \wedge \\
dom\ voucherID = dom\ itemQty = dom\ subTotal = cartProductId \wedge \\
dom\ voucherID' = dom\ itemQty' = dom\ subTotal' = cartProductId') \\
\vee \\
(\#shopeeCoin < 10 \wedge rep! = insufficientShopeeCoin \wedge dom\ shopeeCoin = cartUserId \wedge dom\ shopeeCoin' = cartUse \\
\wedge \\
(cartProdID' = cartProdID \wedge shopeeCoin' = shopeeCoin \wedge user' = user)) \\
)
\end{array}
$$

6.6 Conclusion

In conclusion, during the lecture, practical and tutorial session I have learned the Z schema language that helps me to complete this assignment. I would like to appreciate the lecture, Ms Azurawati and my tutor, Ms Mazlinda who teaches, guides and corrects my mistakes. In the Z schema language I had learned the operation, error, complete schema and the other symbols in the Z schema that needed to be used in the operation, error and complete schema. This subject is not considered as an easy subject because you will need to think and write all the possible schema that relate to the schema so after this phase the tester will be able to review it to determine if the defects or errors exist in the schema. Not only that, I will also need to use my teammate's schema that we had contributed to the parts of the system and finally we integrate it. So, the system schema can be done more effectively and reduce the time used for the person who would need to do all the schema together. Lastly, I would like to say thank you to my teammate who can do great teamwork and punctually hand in the task we assigned to avoid delaying the other people's work.

7.0 Payment Module

7.1 State Space Schema

Payment

$$
\begin{array}{l}
paymentId : \mathbb{P}\ PAYMENTID \\
paymentStatus : \mathbb{P}\ (PAYMENTID \pfun PAYMENTSTATUS) \\
paymentMethod : \mathbb{P}\ (PAYMENTID \pfun PAYMENTMETHOD) \\
paymentDate : \mathbb{P}\ (PAYMENTID \pfun PAYMENTDATE) \\
paymentAmount : (PAYMENTID \pfun \mathbb{N}) \\
delivery : (PAYMENTID \pfun DELIVERYADDRESS) \\
paymentUserId : (PAYMENTID \pfun USERID) \\
paymentCartId : (PAYMENTID \pfun CARTID) \\
paymentProductId : (PAYMENTID \pfun PROUDUCTID) \\
\hline
\#paymentAmount \geqslant maxPaymentAmount \\
dom\ paymentProductId = dom\ paymentCartId = dom\ paymentUserId = dom\ delivery = \\
dom\ paymentAmount = dom\ paymentDate = dom\ paymentMethod = dom\ paymentStatus = \\
paymentId
\end{array}
$$

7.2 Initial State Schema

InitPayment

Payment
paymenId = ∅
paymentStatus = ∅
paymentMethod = ∅
paymentDate = ∅
paymentAmount = 0
delivery = ∅
paymentUserId = ∅
paymentCartId = ∅
paymentProductId = ∅

7.3 Operation Schema

MakePayment

Δ *Payment*

Ξ *User*

Ξ *Cart*

paymentId? : PAYMENTID

paymentMethod? : PAYMENTMETHOD

userId? : *USERID*

cartId? *: CARTID*

deliveryAddress? : *DELIVERYADDRESS*

---

paymentMethod? $\in$ {*paymentMethod*}

userStatus(|{*user?*}|) = *loggedIn*

userId $\in$ *dom paymentUserId*

cartId $\in$ *dom paymentCartId*

deliveryAddress? $\in$ *dom delivery*

#*paymentAmount* $\leqslant$ *maxPaymentAmount*

paymentStatus' = *paymentStatus* $\cup$ {*paymentId* $\mapsto$ *'successful'*}

cart' = *cart* $\setminus$ {*cartId?*}

ViewPaymentStatus

Ξ *Payment*

Ξ *User*

paymentId? : *PAYMENTID*

---

paymentId $\in$ {*paymentId*}

userStatus(|{*user?*}|) = *loggedIn*

paymentStatus' = (*paymentMethod* $\notin$ {*paymentMethod*} $\lor$ *paymentUserId* $\notin$ *userId*) $\oplus$ *unsuccessful*

paymentStatus' = (*paymentMethod* $\in$ {*paymentMethod*} $\land$ *paymentUserId* $\in$ *userId*) $\oplus$ *successful*

AddPaymentMethod

Δ *Payment*

Ξ *User*

paymentId? : *PAYMENTID*

newPaymentMethod? : PAYMENTMETHOD

---

paymentId? ∈ dom paymentAmount

userStatus(|{user?}|) = loggedIn

newPaymentMethod? ∈ paymentMethod

{newPaymentMethod?} ∉ ran paymentMethod

paymentMethod' = paymentMethod ∪ {paymentId ↦ newPaymentMethod?}

ChangeDeliveryAddress

Δ *Payment*

Ξ *User*

paymentId? : *PAYMENTID*

newDeliveryAddress? : *DELIVERYADDRESS*

---

userStatus (|{user?}|) = loggedIn

paymentId ∈ dom paymentAmount

{*newDeliveryAddress*?} ∉ *dom delivery*

paymentId ∈ dom paymentUserId

newDeliveryAddress ∈ dom delivery

delivery' = (*delivery* ⩤ {*paymentId* ↦ *delivery*}) ∪ {*paymentId* ↦ *newDeliveryAddress*}

7.4 Error Scenarios

Error Scenario:

Schema Name	Success Pre-Condition	Failure Pre-Condition	Remark
MakePayment	*paymentMethod? ∈ {paymentMethod}* *userStatus(|{user?}|) = loggedIn* *userId ∈ dom paymentUserId* *cartId ∈ dom paymentCartId* *deliveryAddress? ∈ dom delivery* *#paymentAmount ≤ maxPaymentAmount*	*paymentMethod? ∉ {paymentMethod}* *userStatus(|{user?}|) ≠ loggedIn* *userId ∉ dom paymentUserId* *cartId ∉ dom paymentCartId* *{deliveryAddress?} ∉ dom delivery* *#paymentAmount > maxPaymentAmount*	1. Invalid Payment Method 2. User not logged In 3. Invalid User 4. Cart does not have products 5. Invalid Address 6. payment amount is out of acceptable limits.
ViewPaymentStatus	*paymentId ∈ {paymentId}* *userStatus(|{user?}|) = loggedIn*	*paymentId ∉ {paymentId}* *userStatus(|{user?}|) ≠ loggedIn*	1. Invalid Payment Id 2. User not logged in
AddPaymentMethod	*paymentId? ∈ dom paymentAmount* *userStatus(|{user?}|) = loggedIn* *newPaymentMethod? ∈ paymentMethod* *{newPaymentMethod?} ∉ ran paymentMethod*	*paymentId? ∉ dom paymentAmount* *userStatus(|{user?}|) ≠ loggedIn* *newPaymentMethod? ∉paymentMethod* *{newPaymentMethod?} ∈ ran paymentMethod*	1. Payment Id Not Exist 2. User not logged In 3. Invalid Payment Method 4. Payment Method Already Exist
ChangeDeliveryAddress	*userStatus (|{user?}|) = loggedIn* *paymentId ∈ dom paymentAmount* *{newDeliveryAddress?} ∉ dom delivery* *paymentId ∈ dom paymentUserId* *newDeliveryAddress ∈ dom delivery*	*userStatus (|{user?}|) ≠ loggedIn* *paymentId ∉ dom paymentAmount* *{newDeliveryAddress?} ∈ dom delivery* *paymentId ∉ dom paymentUserId* *newDeliveryAddress ∉ dom delivery*	1. User not logged In 2. Invalid Payment Id 3. Address Already Exist 4. Invalid User Id 5. Invalid Address

Error Scenario Free Type

RESPONSEPAYMENT:= success | invalidPaymentMethod | userNorLoggedIn | invalid User | emptyCart | invalidAddress | amountIsOutOfLimit invalidPaymentId | paymentIdNotExist | paymentMethodAlreadyExist | addressAlreadyExist

Okay

rep! : *RESPONSEPAYMENT*

---

rep! = *success*

Error Scenario

MakePaymentError

Ξ Payment
Ξ User
Ξ Cart
paymentId? : PAYMENTID
paymentMethod? : PAYMENTMETHOD
userId? : USERID
cartId? : CARTID
deliveryAddress? : DELIVERYADDRESS
rep! : RESPONSEPAYMENT

(paymentMethod? ∉ {paymentMethod} ∧ *rep*! = *invalidPaymentMethod*)
∨
(userStatus⦇{user?}⦈ ≠loggedIn ∧ *rep*! = *userNotLoggedIn*)
∨
(userId ∉ dom paymentUserId ∧ *rep*! = *invalidUser*)
∨
(cartId ∉ dom paymentCartId ∧ *rep*! = *emptyCart*)
∨
({deliveryAddress?} ∉ dom delivery ∧ *rep*! = *invalidAddress*)
∨
(#paymentAmount > maxPaymentAmount ∧ *rep*! = *amountIsOutOfLimit*)

ViewPaymentStatusError

Ξ *Payment*

Ξ *User*

paymentId? : *PAYMENTID*

rep! : *RESPONSEPAYMENT*

(paymentId $\notin$ {paymentId} $\land$ *rep*! = *invalidPaymentId*)

$\lor$

(userStatus$(\!|${user?}$|\!)$ $\neq$ loggedIn $\land$ *rep*! = *userNotLoggedIn*)

AddPaymentMethodError

Ξ *Payment*

Ξ *User*

paymentId? : *PAYMENTID*

newPaymentMethod? : PAYMENTMETHOD

rep! : *RESPONSEPAYMENT*

(*paymentId*? $\notin$ *dom paymentAmount* $\land$ *rep*! = *paymentIdNotExist*)

$\lor$

(*userStatus*$(\!|\{user?\}|\!)$ $\neq$ *loggedIn* $\land$ *rep*! = *userNotLoggedIn*)

$\lor$

(*newPaymentMethod*? $\notin$ *paymentMethod* $\land$ *rep*! = *invalidPaymentMethod*)

$\lor$

({*newPaymentMethod*?} $\in$ *ran paymentMethod* $\land$ *rep*! = *paymentMethodAlreadyExist*)

ChangeDeliveryAddressError

Ξ *Payment*

Ξ *User*

paymentId? : *PAYMENTID*

newDeliveryAddress? : *DELIVERYADDRESS*

rep! : *RESPONSEPAYMENT*

(*userStatus* (|{*user?*}|) $\neq$ *loggedIn* $\land$ *rep*! = *userNotLoggedIn*)

$\lor$

(*paymentId* $\notin$ *dom paymentAmount* $\land$ *rep*! = *invalidPaymentId*)

$\lor$

({*newDeliveryAddress?*} $\in$ *dom delivery* $\land$ *rep*! = *addressAlreadyExist*)

$\lor$

(*paymentId* $\notin$ *dom paymentUserId* $\land$ *rep*! = *invalidUser*)

$\lor$

(*newDeliveryAddress* $\notin$ *dom delivery* $\land$ *rep*! = *invalidAddress*)

7.5 Complete Schema

MakePaymentComplete ≙ (MakePayment ∧ Okay) ∨ MakePaymentError

MakePaymentComplete

$paymentId, paymentId' : \mathbb{P}\ PAYMENTID$
$paymentStatus, paymentStatus' : \mathbb{P}\ (PAYMENTID \pfun PAYMENTSTATUS)$
$paymentMethod, paymentMethod' : \mathbb{P}\ (PAYMENTID \pfun PAYMENTMETHOD)$
$paymentDate, paymentDate' : \mathbb{P}\ (PAYMENTID \pfun PAYMENTDATE)$
$paymentAmount, paymentAmount' : (PAYMENTID \pfun \mathbb{N})$
$delivery, delivery' : (PAYMENTID \pfun DELIVERYADDRESS)$
$paymentUserId, paymentUserId' : (PAYMENTID \pfun USERID)$
$paymentCartId, paymentCartId' : (PAYMENTID \pfun CARTID)$
$paymentProductId, paymentProductId' : (PAYMENTID \pfun PROUDUCTID)$

$userId, userId' : \mathbb{P}\ USERID$
$userName, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone, userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$cartId, cartId' : \mathbb{P}\ CARTID$
$cartProductId, cartProductId' : \mathbb{P}\ (CARTID \pfun PRODID)$

$cartUserId, cartUserId' : \mathbb{P}(CARTID \pfun USERID)$

$paymentId? : PAYMENTID$
$paymentMethod? : PAYMENTMETHOD$
$userId? : USERID$
$cartId? : CARTID$
$deliveryAddress? : DELIVERYADDRESS$
$rep! : RESPONSEPAYMENT$

$paymentMethod? \in \{paymentMethod\} \land userStatus(\!|\{user?\}|\!) = loggedIn \land userId \in dom\ paymentUserId$
$\land\ cartId \in dom\ paymentCartId \land deliveryAddress? \in dom\ delivery \land$
$\#paymentAmount \leqslant maxPaymentAmount \land$
$paymentStatus' = paymentStatus \cup \{paymentId \mapsto 'successful'\} \land cart' = cart \setminus \{cartId?\} \land$
$rep! = success$

$\#paymentAmount \geqslant maxPaymentAmount \land \#paymentAmount' \geqslant maxPaymentAmount \land$
$dom\ paymentProductId = dom\ paymentCartId = dom\ paymentUserId = dom\ delivery =$
$dom\ paymentAmount = dom\ paymentDate = dom\ paymentMethod = dom\ paymentStatus = paymentId \land$
$dom\ paymentProductId' = dom\ paymentCartId' = dom\ paymentUserId' = dom\ delivery' =$
$dom\ paymentAmount' = dom\ paymentDate' = dom\ paymentMethod' = dom\ paymentStatus' = paymentId'$
$dom\ userName = dom\ userEmail = dom\ userGender = dom\ userPhone = dom\ password =$
$dom\ userHistory = dom\ userStatus = dom\ userId \land$
$dom\ userName' = dom\ userEmail' = dom\ userGender' = dom\ userPhone' = dom\ password' =$
$dom\ userHistory' = dom\ userStatus' = dom\ userId'$
$dom\ cartUserId = dom\ cartProductId = dom\ cartId \land$
$dom\ cartUserId' = dom\ cartProductId' = dom\ cartId'$
$\lor$
(
$(paymentMethod? \notin \{paymentMethod\} \land rep! = invalidPaymentMethod \land$
$dom\ paymentProductId = dom\ paymentCartId = dom\ paymentUserId = dom\ delivery =$
$dom\ paymentAmount = dom\ paymentDate = dom\ paymentMethod = dom\ paymentStatus = paymentId \land$
$dom\ paymentProductId' = dom\ paymentCartId' = dom\ paymentUserId' = dom\ delivery' =$
$dom\ paymentAmount' = dom\ paymentDate' = dom\ paymentMethod' = dom\ paymentStatus' = paymentId'$
$\lor$
$(userStatus(\!|\{user?\}|\!) \neq loggedIn \land rep! = userNotLoggedIn \land$
$dom\ userName = dom\ userEmail = dom\ userGender = dom\ userPhone = dom\ password =$
$dom\ userHistory = dom\ userStatus = dom\ userId \land$
$dom\ userName' = dom\ userEmail' = dom\ userGender' = dom\ userPhone' = dom\ password' =$
$dom\ userHistory' = dom\ userStatus' = dom\ userId'$
$\lor$
$(userId \notin dom\ paymentUserId \land rep! = invalidUser \land$
$dom\ paymentProductId = dom\ paymentCartId = dom\ paymentUserId = dom\ delivery =$

$$
\begin{aligned}
&\mathit{dom\ paymentAmount} = \mathit{dom\ paymentDate} = \mathit{dom\ paymentMethod} = \mathit{dom\ paymentStatus} = \mathit{paymentId} \land \\
&\mathit{dom\ paymentProductId'} = \mathit{dom\ paymentCartId'} = \mathit{dom\ paymentUserId'} = \mathit{dom\ delivery'} = \\
&\mathit{dom\ paymentAmount'} = \mathit{dom\ paymentDate'} = \mathit{dom\ paymentMethod'} = \mathit{dom\ paymentStatus'} = \mathit{paymentId'} \\
&\lor \\
&(\mathrm{cartId} \notin \mathrm{dom\ paymentCartId} \land \mathit{rep}! = \mathit{emptyCart} \land \\
&\mathit{dom\ paymentProductId} = \mathit{dom\ paymentCartId} = \mathit{dom\ paymentUserId} = \mathit{dom\ delivery} = \\
&\mathit{dom\ paymentAmount} = \mathit{dom\ paymentDate} = \mathit{dom\ paymentMethod} = \mathit{dom\ paymentStatus} = \mathit{paymentId} \land \\
&\mathit{dom\ paymentProductId'} = \mathit{dom\ paymentCartId'} = \mathit{dom\ paymentUserId'} = \mathit{dom\ delivery'} = \\
&\mathit{dom\ paymentAmount'} = \mathit{dom\ paymentDate'} = \mathit{dom\ paymentMethod'} = \mathit{dom\ paymentStatus'} = \mathit{paymentId'} \\
&\lor \\
&(\{\mathrm{deliveryAddress?}\} \notin \mathrm{dom\ delivery} \land \mathit{rep}! = \mathit{invalidAddress} \land \\
&\mathit{dom\ paymentProductId} = \mathit{dom\ paymentCartId} = \mathit{dom\ paymentUserId} = \mathit{dom\ delivery} = \\
&\mathit{dom\ paymentAmount} = \mathit{dom\ paymentDate} = \mathit{dom\ paymentMethod} = \mathit{dom\ paymentStatus} = \mathit{paymentId} \land \\
&\mathit{dom\ paymentProductId'} = \mathit{dom\ paymentCartId'} = \mathit{dom\ paymentUserId'} = \mathit{dom\ delivery'} = \\
&\mathit{dom\ paymentAmount'} = \mathit{dom\ paymentDate'} = \mathit{dom\ paymentMethod'} = \mathit{dom\ paymentStatus'} = \mathit{paymentId'} \\
&\lor \\
&(\#\mathrm{paymentAmount} > \mathrm{maxPaymentAmount} \land \mathit{rep}! = \mathit{amountIsOutOfLimit} \land \\
&\#\mathit{paymentAmount} \geqslant \mathit{maxPaymentAmount} \land \#\mathit{paymentAmount'} \geqslant \mathit{maxPaymentAmount} \\
&\land \\
&(\mathit{paymentId'} = \mathit{paymentId} \land \mathit{paymentMethod'} = \mathit{paymentMethod} \land \mathit{userId'} = \mathit{userId} \land \mathit{cartId'} = \mathit{cartId} \land \\
&\mathit{deliveryAddress'} = \mathit{deliveryAddress}) \\
&)
\end{aligned}
$$

ViewPaymentStatusComplete ≙ (ViewPaymentStatus ∧ Okay) ∨ ViewPaymentStatusError

ViewPaymentStatusComplete

$paymentId, paymentId' : \mathbb{P}\ PAYMENTID$
$paymentStatus, paymentStatus' : \mathbb{P}\ (PAYMENTID \pfun PAYMENTSTATUS)$
$paymentMethod, paymentMethod' : \mathbb{P}\ (PAYMENTID \pfun PAYMENTMETHOD)$
$paymentDate, paymentDate' : \mathbb{P}\ (PAYMENTID \pfun PAYMENTDATE)$
$paymentAmount, paymentAmount' : (PAYMENTID \pfun \mathbb{N})$
$delivery, delivery' : (PAYMENTID \pfun DELIVERYADDRESS)$
$paymentUserId, paymentUserId' : (PAYMENTID \pfun USERID)$
$paymentCartId, paymentCartId' : (PAYMENTID \pfun CARTID)$
$paymentProductId, paymentProductId' : (PAYMENTID \pfun PROUDUCTID)$

$userId, userId' : \mathbb{P}\ USERID$
$userName, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone, userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$paymentId? : PAYMENTID$
$rep! : RESPONSEPAYMENT$

$paymentId \in \{paymentId\} \land userStatus(\!|\{user?\}|\!) = loggedIn \land$
$paymentStatus' = (paymentMethod \notin \{paymentMethod\} \lor paymentUserId \notin userId) \oplus unsuccessful \land$
$paymentStatus' = (paymentMethod \in \{paymentMethod\} \land paymentUserId \in userId) \oplus successful \land$
$rep! = success$

$\#paymentAmount \geqslant maxPaymentAmount \land \#paymentAmount' \geqslant maxPaymentAmount \land$
$dom\ paymentProductId = dom\ paymentCartId = dom\ paymentUserId = dom\ delivery =$
$dom\ paymentAmount = dom\ paymentDate = dom\ paymentMethod = dom\ paymentStatus = paymentId \land$
$dom\ paymentProductId' = dom\ paymentCartId' = dom\ paymentUserId' = dom\ delivery' =$
$dom\ paymentAmount' = dom\ paymentDate' = dom\ paymentMethod' = dom\ paymentStatus' = paymentId'$
$dom\ userName = dom\ userEmail = dom\ userGender = dom\ userPhone = dom\ password =$
$dom\ userHistory = dom\ userStatus = dom\ userId \land$
$dom\ userName' = dom\ userEmail' = dom\ userGender' = dom\ userPhone' = dom\ password' =$
$dom\ userHistory' = dom\ userStatus' = dom\ userId'$
$\lor$
(
((paymentId $\notin$ {paymentId} $\land$ *rep*! = *invalidPaymentId* $\land$

dom paymentProductId = dom paymentCartId = dom paymentUserId = dom delivery =
dom paymentAmount = dom paymentDate = dom paymentMethod = dom paymentStatus = paymentId ∧
dom paymentProductId′ = dom paymentCartId′ = dom paymentUserId′ = dom delivery′ =
dom paymentAmount′ = dom paymentDate′ = dom paymentMethod′ = dom paymentStatus′ = paymentId′
∨
(userStatus⦇{user?}⦈ ≠ loggedIn ∧ *rep*! = *userNotLoggedIn* ∧
dom userName = dom userEmail = dom userGender = dom userPhone = dom password =
dom userHistory = dom userStatus = dom userId ∧
dom userName′ = dom userEmail′ = dom userGender′ = dom userPhone′ = dom password′ =
dom userHistory′ = dom userStatus′ = dom userId′
∧
(*payemntId′= paymentId*)
)

AddPaymentMethodComplete ≙ (AddPaymentMethod ∧ Okay) ∨ AddPaymentMethodError

AddPaymentMethodComplete

$paymentId, paymentId' : \mathbb{P}\ PAYMENTID$
$paymentStatus, paymentStatus' : \mathbb{P}\ (PAYMENTID \pfun PAYMENTSTATUS)$
$paymentMethod, paymentMethod' : \mathbb{P}\ (PAYMENTID \pfun PAYMENTMETHOD)$
$paymentDate, paymentDate' : \mathbb{P}\ (PAYMENTID \pfun PAYMENTDATE)$
$paymentAmount, paymentAmount' : (PAYMENTID \pfun \mathbb{N})$
$delivery, delivery' : (PAYMENTID \pfun DELIVERYADDRESS)$
$paymentUserId, paymentUserId' : (PAYMENTID \pfun USERID)$
$paymentCartId, paymentCartId' : (PAYMENTID \pfun CARTID)$
$paymentProductId, paymentProductId' : (PAYMENTID \pfun PROUDUCTID)$

$userId, userId' : \mathbb{P}\ USERID$
$userName, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone,\ userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$paymentId? : PAYMENTID$
$newPaymentMethod? : PAYMENTMETHOD$
$rep! : RESPONSEPAYMENT$

$paymentId? \in dom\ paymentAmount \land userStatus(\!|\{user?\}|\!) = loggedIn \land$
$newPaymentMethod? \in paymentMethod \land$
$\{newPaymentMethod?\} \notin ran\ paymentMethod \land$
$paymentMethod' = paymentMethod \cup \{paymentId \mapsto newPaymentMethod?\}$

$\#paymentAmount \geqslant maxPaymentAmount \land \#paymentAmount' \geqslant maxPaymentAmount \land$
$dom\ paymentProductId = dom\ paymentCartId = dom\ paymentUserId = dom\ delivery =$
$dom\ paymentAmount = dom\ paymentDate = dom\ paymentMethod = dom\ paymentStatus = paymentId \land$
$dom\ paymentProductId' = dom\ paymentCartId' = dom\ paymentUserId' = dom\ delivery' =$
$dom\ paymentAmount' = dom\ paymentDate' = dom\ paymentMethod' = dom\ paymentStatus' = paymentId'$
$dom\ userName = dom\ userEmail = dom\ userGender = dom\ userPhone = dom\ password =$
$dom\ userHistory = dom\ userStatus = dom\ userId \land$
$dom\ userName' = dom\ userEmail' = dom\ userGender' = dom\ userPhone' = dom\ password' =$
$dom\ userHistory' = dom\ userStatus' = dom\ userId'$
$dom\ cartUserId = dom\ cartProductId = dom\ cartId \land$

$$
\begin{aligned}
&dom\ cartUserId' = dom\ cartProductId' = dom\ cartId' \\
&\lor \\
&(\\
&((paymentId? \notin dom\ paymentAmount \land rep! = paymentIdNotExist \land \\
&dom\ paymentProductId = dom\ paymentCartId = dom\ paymentUserId = dom\ delivery = \\
&dom\ paymentAmount = dom\ paymentDate = dom\ paymentMethod = dom\ paymentStatus = paymentId \land \\
&dom\ paymentProductId' = dom\ paymentCartId' = dom\ paymentUserId' = dom\ delivery' = \\
&dom\ paymentAmount' = dom\ paymentDate' = dom\ paymentMethod' = dom\ paymentStatus' = paymentId' \\
&\lor \\
&(userStatus(\!|\{user?\}|\!) \neq loggedIn \land rep! = userNotLoggedIn \land \\
&dom\ userName = dom\ userEmail = dom\ userGender = dom\ userPhone = dom\ password = \\
&dom\ userHistory = dom\ userStatus = dom\ userId \land \\
&dom\ userName' = dom\ userEmail' = dom\ userGender' = dom\ userPhone' = dom\ password' = \\
&dom\ userHistory' = dom\ userStatus' = dom\ userId' \\
&\lor \\
&(newPaymentMethod? \notin paymentMethod \land rep! = invalidPaymentMethod \land \\
&dom\ paymentProductId = dom\ paymentCartId = dom\ paymentUserId = dom\ delivery = \\
&dom\ paymentAmount = dom\ paymentDate = dom\ paymentMethod = dom\ paymentStatus = paymentId \land \\
&dom\ paymentProductId' = dom\ paymentCartId' = dom\ paymentUserId' = dom\ delivery' = \\
&dom\ paymentAmount' = dom\ paymentDate' = dom\ paymentMethod' = dom\ paymentStatus' = paymentId' \\
&\lor \\
&(\{newPaymentMethod?\} \in ran\ paymentMethod \land rep! = paymentMethodAlreadyExist \land \\
&dom\ paymentProductId = dom\ paymentCartId = dom\ paymentUserId = dom\ delivery = \\
&dom\ paymentAmount = dom\ paymentDate = dom\ paymentMethod = dom\ paymentStatus = paymentId \land \\
&dom\ paymentProductId' = dom\ paymentCartId' = dom\ paymentUserId' = dom\ delivery' = \\
&dom\ paymentAmount' = dom\ paymentDate' = dom\ paymentMethod' = dom\ paymentStatus' = paymentId' \\
&\land \\
&(paymentId' = paymentId \land newPaymentMethod' = newPaymentMethod) \\
&)
\end{aligned}
$$

ChangeDeliveryAddressComplete ≙ (ChangeDeliveryAddress ∧ Okay) ∨

ChangeDeliveryAddressError

ChangeDeliveryAddressComplete

$$
\begin{array}{l}
paymentId, paymentId' : \mathbb{P}\ PAYMENTID \\
paymentStatus, paymentStatus' : \mathbb{P}\ (PAYMENTID \pfun PAYMENTSTATUS) \\
paymentMethod, paymentMethod' : \mathbb{P}\ (PAYMENTID \pfun PAYMENTMETHOD) \\
paymentDate, paymentDate' : \mathbb{P}\ (PAYMENTID \pfun PAYMENTDATE) \\
paymentAmount, paymentAmount' : (PAYMENTID \pfun \mathbb{N}) \\
delivery, delivery' : (PAYMENTID \pfun DELIVERYADDRESS) \\
paymentUserId, paymentUserId' : (PAYMENTID \pfun USERID) \\
paymentCartId, paymentCartId' : (PAYMENTID \pfun CARTID) \\
paymentProductId, paymentProductId' : (PAYMENTID \pfun PROUDUCTID) \\
\\
userId, userId' : \mathbb{P}\ USERID \\
userName, userName' : \mathbb{P}\ (USERID \times USERNAME) \\
userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER) \\
userPhone,\ userPhone' : \mathbb{P}\ (USERID \times USERPHONE) \\
userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD) \\
userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY) \\
userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS) \\
\\
paymentId? : PAYMENTID \\
newDeliveryAddress? : DELIVERYADDRESS \\
rep! : RESPONSEPAYMENT \\
\hline
userStatus\ (\!|\{user?\}|\!) = loggedIn \land paymentId \in dom\ paymentAmount \\
\land\ \{newDeliveryAddress?\} \notin dom\ delivery \land paymentId \in dom\ paymentUserId \\
\land\ newDeliveryAddress \in dom\ delivery \\
\land\ delivery' = (delivery \setminus \{paymentId \mapsto delivery\}) \cup \{paymentId \mapsto newDeliveryAddress\} \\
\\
\#paymentAmount \geqslant maxPaymentAmount \land \#paymentAmount' \geqslant maxPaymentAmount \land \\
dom\ paymentProductId = dom\ paymentCartId = dom\ paymentUserId = dom\ delivery = \\
dom\ paymentAmount = dom\ paymentDate = dom\ paymentMethod = dom\ paymentStatus = paymentId \land \\
dom\ paymentProductId' = dom\ paymentCartId' = dom\ paymentUserId' = dom\ delivery' = \\
dom\ paymentAmount' = dom\ paymentDate' = dom\ paymentMethod' = dom\ paymentStatus' = paymentId' \\
dom\ userName = dom\ userEmail = dom\ userGender = dom\ userPhone = dom\ password = \\
dom\ userHistory = dom\ userStatus = dom\ userId \land \\
dom\ userName' = dom\ userEmail' = dom\ userGender' = dom\ userPhone' = dom\ password' =
\end{array}
$$

$$
\begin{aligned}
&dom\ userHistory' = dom\ userStatus' = dom\ userId' \\
&dom\ cartUserId = dom\ cartProductId = dom\ cartId \land \\
&dom\ cartUserId' = dom\ cartProductId' = dom\ cartId' \\
&\lor \\
&(\\
&((userStatus\ (\!|\{user?\}|\!) \neq loggedIn \land rep! = userNotLoggedIn \land \\
&dom\ userName = dom\ userEmail = dom\ userGender = dom\ userPhone = dom\ password = \\
&dom\ userHistory = dom\ userStatus = dom\ userId \land \\
&dom\ userName' = dom\ userEmail' = dom\ userGender' = dom\ userPhone' = dom\ password' = \\
&dom\ userHistory' = dom\ userStatus' = dom\ userId' \\
&\lor \\
&(paymentId \notin dom\ paymentAmount \land rep! = invalidPaymentId \land \\
&dom\ paymentProductId = dom\ paymentCartId = dom\ paymentUserId = dom\ delivery = \\
&dom\ paymentAmount = dom\ paymentDate = dom\ paymentMethod = dom\ paymentStatus = paymentId \land \\
&dom\ paymentProductId' = dom\ paymentCartId' = dom\ paymentUserId' = dom\ delivery' = \\
&dom\ paymentAmount' = dom\ paymentDate' = dom\ paymentMethod' = dom\ paymentStatus' = paymentId' \\
&\lor \\
&(\{newDeliveryAddress?\} \in dom\ delivery \land rep! = addressAlreadyExist \land \\
&dom\ paymentProductId = dom\ paymentCartId = dom\ paymentUserId = dom\ delivery = \\
&dom\ paymentAmount = dom\ paymentDate = dom\ paymentMethod = dom\ paymentStatus = paymentId \land \\
&dom\ paymentProductId' = dom\ paymentCartId' = dom\ paymentUserId' = dom\ delivery' = \\
&dom\ paymentAmount' = dom\ paymentDate' = dom\ paymentMethod' = dom\ paymentStatus' = paymentId' \\
&\lor \\
&(paymentId \notin dom\ paymentUserId \land rep! = invalidUser \land \\
&dom\ paymentProductId = dom\ paymentCartId = dom\ paymentUserId = dom\ delivery = \\
&dom\ paymentAmount = dom\ paymentDate = dom\ paymentMethod = dom\ paymentStatus = paymentId \land \\
&dom\ paymentProductId' = dom\ paymentCartId' = dom\ paymentUserId' = dom\ delivery' = \\
&dom\ paymentAmount' = dom\ paymentDate' = dom\ paymentMethod' = dom\ paymentStatus' = paymentId' \\
&\lor \\
&(newDeliveryAddress \notin dom\ delivery \land rep! = invalidAddress \land \\
&dom\ paymentProductId = dom\ paymentCartId = dom\ paymentUserId = dom\ delivery = \\
&dom\ paymentAmount = dom\ paymentDate = dom\ paymentMethod = dom\ paymentStatus = paymentId \land \\
&dom\ paymentProductId' = dom\ paymentCartId' = dom\ paymentUserId' = dom\ delivery' = \\
&dom\ paymentAmount' = dom\ paymentDate' = dom\ paymentMethod' = dom\ paymentStatus' = paymentId' \\
&\land \\
&(paymentId' = paymentId \land newDeliveryAddress' = newDeliveryAddress) \\
&)
\end{aligned}
$$

7.6 Conclusion

In concluding this extensive Z schema project, I would like to express my heartfelt gratitude to several individuals who have played pivotal roles in its successful completion. First and foremost, I want to extend my deepest appreciation to my dedicated and knowledgeable tutor, Ms. Mazlinda. Her unwavering support, insightful guidance, and willingness to patiently address my questions and concerns were instrumental in navigating the complexities of formal specification with confidence and precision. I would also like to extend my gratitude to my lecturer, Ms. Azurawati, whose expertise and comprehensive course materials laid the foundation for our understanding of formal methods and their practical application. Her commitment to our learning experience has been invaluable. Additionally, I would like to acknowledge and thank my teammates, who collaborated tirelessly throughout this project. Their teamwork, dedication, and collective efforts in refining the schemas and ensuring their correctness were pivotal to the project's success. This project has not only deepened my knowledge of formal specification but has also honed my collaborative and problem-solving skills. As I reflect on this journey, I am grateful for the support and contributions of these remarkable individuals who have been pivotal in achieving our project's goals.

8.0 ShopeeFood Module

8.1 State Space Schema

Restuarant

$restId : \mathbb{P}\ RESTID$
$restName : RESTID \pfun RESTNAME$
$restInfo : RESTID \pfun RESTINFO$
$restRate : RESTID \pfun RESTRATE$

$\text{dom}\ restInfo = \text{dom}\ restRate = restId$
$minRate \leqslant restRate \geqslant maxRate$

Food

$Restuarant$
$foodId : RESTID \pfun RESTFOODID$
$foodName : RESTFOODID \pfun RESTFOODNAME$
$foodPrice : RESTFOODID \pfun RESTFOODPRICE$
$foodDes : RESTFOODID \pfun RESTFOODDES$

$\text{dom}\ foodId = restId$
$\text{dom}\ foodName = \text{dom}\ foodPrice = \text{dom}\ foodDes = foodId$

Basket

$User$
$basketId : USERID \pfun FOODBASKETID$
$totalQty : FOODBASKETID \pfun TOTALFOODQTY$
$totalFoodPrice : FOODBASKETID \pfun TOTALFOODPRICE$

$\text{dom}\ basketId = userId$

BasketItem

$itemId : FOODBASKETID \pfun BASKETITEMID$
$basketFood : BASKETITEMID \pfun RESTFOODID$
$foodQty : BASKETITEMID \pfun FOODQTY$

$foodQty \geqslant minFoodQty$

FavPlace

$User$
$favPlaceId : USERID \pfun FAVPLACEID$
$favPlace : FAVPLACEID \pfun RESTID$

$\text{dom}\ favPlaceId = userId$

FoodOrder

User

Basket

$foodOrderId : USERID \pfun FOODORDERID$

$orderBasket : FOODORDERID \pfun BASKETID$

$orderState : FOODORDERID \pfun ORDERSTATE$

$orderDate : FOODORDERID \pfun ORDERDATE$

$orderPayment : FOODORDERID \pfun ORDERPAYMENT$

$orderRate : FOODORDERID \pfun ORDERRATE$

$\mathrm{dom}\ foodOrderId = userId$

$\mathrm{ran}\ orderBasket = basketId$

8.2 Initial State Schema

InitRestuarant

Restuarant

restId = ∅
restName = ∅
restInfo = ∅
restRate = 0

InitFood

Food

restId = ∅
restName = ∅
restInfo = ∅
restRate = 0

foodId = ∅
foodName = ∅
foodPrice = ∅
foodDes = ∅

InitBasket

Basket

basketId = ∅
totalQty = 0
totalFoodPrice = 0

userId = ∅
userName = ∅
userEmail = ∅
userGender = *male*
userPhone = ∅
password = ∅
userHistory = ∅
userStatus = *loggedOut*

InitBasketItem

BasketItem

itemId = ∅
basketFood = ∅
foodQty = 0

InitFavPlace

FavPlace

favPlaceId = ∅
favPlace = ∅

userId = ∅
userName = ∅
userEmail = ∅
userGender = *male*
userPhone = ∅
password = ∅
userHistory = ∅
userStatus = *loggedOut*

InitFoodOrder

FoodOrder
foodOrderId = ∅ *orderBasket* = ∅ *orderState* = *processing* *orderDate* = ∅ *orderPayment* = *credit card* *orderRate* = 0 *orderDate* = ∅ *basketId* = ∅ *totalQty* = 0 *totalFoodPrice* = 0 *userId* = ∅ *userName* = ∅ *userEmail* = ∅ *userGender* = *male* *userPhone* = ∅ *password* = ∅ *userHistory* = ∅ *userStatus* = *loggedOut*

8.3 Operation Schema

Operation 1: Create, Retrieve, Update & Delete Food Basket

AddProductIntoBasket

Δ *Basket*
Δ *BasketItem*
Ξ *Food*

userId? : *USERID*
itemId? : *BASKETITEMID*
foodId? : *RESTFOODID*
quantity? : $\mathbb{N}$

basket : *FOODBASKETID*
total : $\mathbb{N}$
price : $\mathbb{N}$
totalPrice : $\mathbb{N}$

(*userId*? $\in$ *userId* $\wedge$ *userStatus*($|${*userId*?}$|$) = *loggedIn*)
userId? $\in$ {*userId*?} $\lhd$ *basketId*
foodId? $\in$ *foodId* $\rhd$ {*foodId*?}
quantity? $\geqslant$ 1

basket = *basketId* ($|${*userId*?}$|$)
total = {*basket*} $\lhd$ *totalQty*
total = *total* + *quantity*?

price = {*foodId*} $\lhd$ *foodPrice*
totalPrice = {*basket*} $\lhd$ *totalFoodPrice*
totalPrice = *totalPrice* + (*price* * *quantity*?)

itemId′ = *itemId* $\cup$ {*basketId* $\mapsto$ *itemId*?}
basketFood′ = *itemId* $\cup$ {*itemId*? $\mapsto$ *foodId*?}
foodQty′ = *itemId* $\bigcup$ {*itemId*? $\mapsto$ *quantity*?}

basketId′ = *basketId*
totalQty′ = *totalQty* $\oplus$ {*basket* $\mapsto$ *total*}
totalFoodPrice′ = *totalFoodPrice* $\oplus$ {*basket* $\mapsto$ *totalPrice*}

ViewAllBasketProduct

Ξ *Basket*
Ξ *BasketItem*
Ξ *Food*
userId? : *USERID*
basketId? : *FOODBASKETID*

restId! : *RESTFOODID*
restName! : *RESTNAME*
basketItemId! : $\mathbb{P}$ *BASKETITEMID*
basketFood! : $\mathbb{P}$ *RESTFOODID*
basketfoodId : $\mathbb{P}$ *RESTFOODID*
foodName! : $\mathbb{P}$ *RESTFOODNAME*
foodPrice! : $\mathbb{N}$
foodQty! : $\mathbb{N}$
totalPrice! : $\mathbb{N}$
totalQty! : $\mathbb{N}$

(*userId*? $\in$ *userId* $\wedge$ *userStatus*($\{$*userId*?$\}$) = *loggedIn*)
userId? $\in$ {*userId*?} $\lhd$ *basketId*
basketId? $\in$ {*userId*?} $\lhd$ *basketId*

basketItemId! = *itemId* ($\{$*basketId*?$\}$)
basketFood! = *basketFood* ($\{$ *itemId* ($\{$*basketId*?$\}$) $\}$)
basketfoodId = *basketFood* ($\{$ *itemId* ($\{$*basketId*?$\}$) $\}$)
restId! = *foodId* $\rhd$ { *basketfoodId* }
restName! = *restName* ($\{$ *foodId* $\rhd$ { *basketfoodId* }$\}$)
foodName! = *foodId* ($\{$ *basketfoodId*$\}$)
foodPrice! = *foodPrice* ($\{$ *basketfoodId* $\}$)
foodQty! = *foodQty* ($\{$ *basketfoodId* $\}$)

totalPrice! = *totalFoodPrice* ($\{$*basketId*?$\}$)
totalQty! = *totalQty* ($\{$*basketId*?$\}$)

UpdateBasketProduct

Δ *Basket*
Δ *BasketItem*
userId? : *USERID*
itemId? : *ITEMID*
quantity? : $\mathbb{N}$

basket : *FOODBASKETID*
totalprice : $\mathbb{N}$
price : *N*
totalqty : $\mathbb{N}$
qty : $\mathbb{N}$

(*userId*? $\in$ *userId* $\wedge$ *userStatus*($\{$*userId*?$\}$) = *loggedIn*)
userId? $\in$ $\{$*userId*?$\}$ $\lhd$ *basketId*

basket = *basketId* ($\{$*userId*?$\}$)
itemId? $\in$ *itemId* ($\{$*basketId*$\}$)
quantity $\geqslant$ 1

itemId′ = *itemId*
basketFood = *basketFood*
qty = *foodQty* ($\{$*itemId*?$\}$)
foodQty′ = *foodQty* $\oplus$ $\{$*itemId*? $\mapsto$ *quantity*?$\}$

totalqty = *totalQty* ($\{$*basket*$\}$)
totalprice = *totalFoodPrice* ($\{$*basket*$\}$)
price = *foodPrice* ($\{$ $\{$*itemId*?$\}$ $\lhd$ *basketFood* $\}$)

basketId′ = *basketId*
totalQty′ = *totalQty* $\oplus$ { *basket* $\mapsto$ (*totalqty* – *qty* + *quantity*?)}
totalFoodPrice′ = *totalFoodPrice* $\oplus$ { *basket* $\mapsto$ (*totalprice* – (*price***qty*) + (*price***quantity*) }

RemoveBasketProduct

Ξ *Basket*
Ξ *BasketItem*
userId? : *USERID*
itemId? : *BASKETITEMID*

basket : *FOODBASKETID*
total : $\mathbb{N}$
qty : $\mathbb{N}$
totalPrice : $\mathbb{N}$
price : $\mathbb{N}$

(*userId*? $\in$ *userId* $\wedge$ *userStatus*($\{$*userId*?$\}$) = *loggedIn*)
userId? $\in$ $\{$*userId*?$\}$ $\lhd$ *basketId*
basket = *basketId* ($\{$*userId*?$\}$)
itemId $\in$ *itemId* ($\{$*basket*$\}$)

total = *totalQty* ($\{$*basket*$\}$)
qty = *foodQty* ($\{$*itemId*?$\}$)
totalPrice = *totalFoodPrice* ($\{$*basket*$\}$)
price = *foodPrice* (*basketFood* ($\{$*itemId*?$\}$))

itemId′ = *itemID* $\lhd$ $\{$*itemId*?$\}$
basketFood′ = $\{$*itemId*?$\}$ $\lhd$ *basketFood*
foodQty′ = $\{$*itemId*?$\}$ $\lhd$ *foodQty*

basketId′ = *basketId*
totalQty′ = *totalQty* $\oplus$ $\{$*basket* $\mapsto$ (*total* - *qty*)$\}$
totalFoodPrice′ = *totalFoodPrice* $\oplus$ $\{$*basket* $\mapsto$ (*totalPrice* – (*price***qty*))$\}$

Operation 2: Create, Retrieve & Delete Favorite Places

AddFavorite

Δ *FavPlace*
Ξ *Restaurant*

userId? : *USERID*
favId : *FAVPLACEID*
restId? : *RESTID*

(*userId*? $\in$ *userId* $\wedge$ *userStatus*(|{*userId*?}|) = *loggedIn*)
restId? $\in$ *restId*
#{*favPlaceId* (|{*userId*?}|)} < 100
favId = *favPlaceId* (|{*userId*?}|)
restId? $\notin$ *favPlace* (|{*favId*}|)

favPlace′ = *favPlace* $\cup$ {*favId* $\mapsto$ *restId*?}

ViewAllFavorite

Ξ *FavPlace*
Ξ*Restuarant*
userId? : *USERID*

favPlaceId! : *FAVPLACEID*
rest : $\mathbb{P}$ *RESTID*
restName! : $\mathbb{P}$ *RESTNAME*
restRate! : $\mathbb{P}$ *RESTRATE*

(*userId*? $\in$ *userId* $\wedge$ *userStatus*(|{*userId*?}|) = *loggedIn*)

favPlaceId! = *favPlaceId* (|{*userId*?}|)
rest = *favPlace* (|{ *favPlaceId* (|{*userId*?}|) }|)
restName! = *restName* (|{*rest*}|)
restRate! = *restRate* (|{*rest*}|)

RemoveFavorite

Δ *FavPlace*
Ξ *Restaurant*

userId? : *USERID*
favId : *FAVPLACEID*
restId? : *RESTID*

(*userId*? $\in$ *userId* $\wedge$ *userStatus*(|{*userId*?}|) = *loggedIn*)
favId $\in$ *favPlaceId* (|{*userId*?}|)
restId? $\in$ *restId*
restId? $\in$ *favPlace* (|{*favId*}|)

favPlace' = *favPlace* $\setminus$ {*favId*? $\mapsto$ *restId*?}

Operation 3: View Order

ViewAllOrder

Ξ *FoodOrder*
userId? : *USERID*

foodOrderId! : $\mathbb{P}$ *FOODORDERID*
order : $\mathbb{P}$ *FOODORDERID*
basketId : $\mathbb{P}$ *FOODBASKETID*
totalQty! : $\mathbb{P}$ *TOTALFOODQTY*
totalFoodPrice! : $\mathbb{P}$ *TOTALFOODPRICE*
state! : $\mathbb{P}$ *ORDERSTATE*
payment! : $\mathbb{P}$ *ORDERPAYMENT*
date! : $\mathbb{P}$ *ORDERDATE*

(*userId*? $\in$ *userId* $\wedge$ *userStatus*(|{*userId*?}|) = *loggedIn*)

order = *foodOrderId* (|{*userId*?}|)
basketId = *basketId* (|{*order*}|)

foodOrderId!= *order*
totalQty! = *totalQty* (|{*basketId*}|)
totalFoodPrice! = *totalFoodPrice* (|{*basketId*}|)
state! = *orderState* (|{*order*}|)
payment! = *orderPayment* (|{*order*}|)
date! = *orderDate* (|{*order*}|)

ViewOrderWithState

Ξ *FoodOrder*
userId? : *USERID*
status? : *ORDERSTATE*

foodOrderId! : $\mathbb{P}$ *FOODORDERID*
order : $\mathbb{P}$ *FOODORDERID*
basketId : $\mathbb{P}$ *FOODBASKETID*
totalQty! : $\mathbb{P}$ *TOTALFOODQTY*
totalFoodPrice! : $\mathbb{P}$ *TOTALFOODPRICE*
state! : $\mathbb{P}$ *ORDERSTATE*
payment! : $\mathbb{P}$ *ORDERPAYMENT*
date! : $\mathbb{P}$ *ORDERDATE*

(*userId*? $\in$ *userId* $\wedge$ *userStatus*($\{$*userId*?$\}$) = *loggedIn*)

order = *foodOrderId* ($\{$*userId*?$\}$) $\wedge$ {*foodOrderId* ($\{$*userId*?$\}$) }$\lhd$ *orderStatus* = *status*?
basketId = *basketId* ($\{$*order*$\}$)

foodOrderId!= *order*
totalQty! = *totalQty* ($\{$*basketId*$\}$)
totalFoodPrice! = *totalFoodPrice* ($\{$*basketId*$\}$)
state! = *orderState* ($\{$*order*$\}$)
payment! = *orderPayment* ($\{$*order*$\}$)
date! = *orderDate* ($\{$*order*$\}$)

RateOrder

Δ *FoodOrder*
Δ *Restaurant*
userId? : *USERID*
foodOrderId? : *FOODORDERID*
restId? : *RESTID*
rating? : $\mathbb{N}$

(*userId*? $\in$ *userId* $\land$ *userStatus*($\{$*userId*?$\}$) = *loggedIn*)

foodOrderId? $\in$ *foodOrderId* ($\{$*userId*?$\}$)
foodState ($\{$*foodOrderId*?$\}$) = *to rate*
restId? $\in$ *restId*
rating = *rating*? $\geqslant$ 0 $\land$ *rating*? $\leqslant$ 5

basketId′ = *basketId*
totalQty′ = *totalQty*
totalFoodPrice′ = *totalFoodPrice*

foodOrderId′ = *foodOrderId*
orderBasket′ = *orderBasket*
orderState′ = *orderStates* $\oplus$ {*foodOrderId* $\mapsto$ *complete*}
orderDate′ = *orderDate*
orderPayment′ = *orderPayment*
orderRate′ = *orderRate* $\oplus$ {*foodOrderId* $\mapsto$ *rating*?}
orderDate′ = *orderDate*

8.4 Error Scenarios

Error Scenario Table

Operation 1: Create, Retrieve, Update & Delete Food Basket

Schema Name	Success Pre-Condition	Failure Pre-Condition	Remark
AddProductIntoBasket	userId? ∈ userId userStatus = loggedIn foodId? ∈ foodId ▷ {foodId?} quantity? > 0	userId? ∉ userId userStatus ≠ loggedIn foodId? ∉ foodId ▷ {foodId?} quantity? ≤ 0	UserNotExist UserNotLogin FoodNotExist InvalidQuantity
ViewAllBasketProduct	userId? ∈ userId userStatus = loggedIn	userId? ∉ userId userStatus ≠ loggedIn	UserNotExist UserNotLogin
UpdateBasketProduct	userId? ∈ userId userStatus = loggedIn itemId? ∈ itemId quantity? > 0	userId? ∉ userId userStatus ≠ loggedIn itemId? ∉ itemId quantity? ≤ 0	UserNotExist UserNotLogin ItemNotExist InvalidQuntity
RemoveBasketProduct	userId? ∈ userId userStatus = loggedIn itemId? ∈ itemId	userId? ∉ userId userStatus ≠ loggedIn itemId? ∉ itemId	UserNotExist UserNotLogin ItemNotExist

Operation 2: Create, Retrieve & Delete Favorite Places

Schema Name	Success Pre-Condition	Failure Pre-Condition	Remark
AddFavorite	userId? ∈ userId userStatus = loggedIn restId? ∈ restId #{favPlaceId ⦇{userId?}⦈} < 100 restId? ∉ favPlace ⦇{favPlaceId ⦇{userId?}⦈}⦈	userId? ∉ userId userStatus ≠ loggedIn restId? ∉ restId #{favPlaceId ⦇{userId?}⦈} < 100 restId? ∈ favPlace ⦇{favPlaceId ⦇{userId?}⦈}⦈	UserNotExist UserNotLogin RestuarantNotExist MaxFavoriteReached ExistFavorite
ViewAllFavorite	userId? ∈ userId userStatus = loggedIn	userId? ∉ userId userStatus ≠ loggedIn	UserNotExist UserNotLogin
RemoveFavorite	userId? ∈ userId userStatus = loggedIn restId? ∈ restId restId? ∈ favPlace ⦇{favPlaceId ⦇{userId?}⦈}⦈	userId? ∉ userId userStatus ≠ loggedIn restId? ∉ restId restId? ∉ favPlace ⦇{favPlaceId ⦇{userId?}⦈}⦈	UserNotExist UserNotLogin RestuarantNotExist FavoriteNotExist

Operation 3: View Order

Schema Name	Success Pre-Condition	Failure Pre-Condition	Remark
ViewAllOrder	userId? ∈ userId userStatus = loggedIn	userId? ∉ userId userStatus ≠ loggedIn	UserNotExist UserNotLogin
ViewOrderWithState	userId? ∈ userId userStatus = loggedIn	userId? ∉ userId userStatus ≠ loggedIn	UserNotExist UserNotLogin
RateOrder	userId? ∈ userId userStatus = loggedIn foodOrderId? ∈ foodOrder ⦇{*userId*?}⦈ restId? ∈ restId orderState = to rate	userId? ∉ userId userStatus ≠ loggedIn foodOrderId? ∉ foodOrder ⦇{*userId*?}⦈ restId? ∉ restId orderState ≠ to rate	UserNotExist UserNotLogin OrderNotExist RestaurantNotExist StateNotToRate

Error Scenario Free Type

RESPONSE ::= success | userNotExist | userNotLogin | foodNotExist | invalidQuntity | itemNotExist | restuarantNotExist | maxFavoriteReached | orderNotExist | invalidRating | stateNotToRate

Success

response! : *RESPONSE*

---

response! = *success*

Error Scenario

Only existing users are able to make operations in ShopeeFood, any user that is not being recorded by the system will not be allowed to make any change in the system.

UserNotExist

Ξ *Basket*
Ξ *FavPlace*
Ξ *FoodOrder*

userId? : *USERID*
response! : *RESPONSE*

---

userId? ∉ *userId*
response! = *userNotExist*

Only logged in user is able make operation in ShopeeFood, any user that is not logged into the system will not allow to make any changes in the system.

UserNotLogin

Ξ *Basket*
Ξ *FavPlace*
Ξ *FoodOrder*

userId? : *USERID*
response! : *RESPONSE*

---

userStatus (|{*userId*?}|) ≠ *loggedIn*
response! = *UserNotLogin*

Non-exsitng food will not be able to added into the basket.

FoodNotExist

Ξ *Food*
foodId? : *RESTFOODID*
response! : *RESPONSE*

---

foodId? $\notin$ *foodId* $\rhd$ {*foodId*?}
response! = *foodNotExist*

When user want to add or update the basket product quantity, a valid number which is more than 0 is required. Otherwise, invalid quantity will not be accepted.

InvalidQuantity

quantity? : $\mathbb{N}$
response! : *RESPONSE*

---

quantity? < 0
response! = *invalidQuantity*

When user want to update the basket product, an existing item is required in the basket. Otherwise, no item will not be updated.

ItemNotExist

Ξ *Basket*
Ξ *BasketItem*
userId? : *USERID*
itemId? : *ITEMID*
response! : *RESPONSE*

---

itemId? $\notin$ *itemId* $(\!|\{$ *basketId*{*userId*?} $\}|\!)$
response = itemNotExist

When a user wants to add a restaurant into their favorite list, an existing restaurant is required. Otherwise, no updates will occur in the list.

RestaurantNotExist

Ξ *Restaurant*
restId? : *RESTID*
response! : *RESPONSE*

---

restId? $\notin$ *restId*
response! = *restaurantNotExist*

The maximum number of restaurants that can be added into the favorite list is 100. If the list has reached the limit, an error message will be shown if the user wants to add more into the list.

MaxFavoriteReached

Ξ *FavPlace*
userId? : *USERID*
response! : *RESPONSE*

$\#(favPlaceId \ (\!|\{userId?\}|\!)) \geqslant 100$
response! = *maxFavoriteReached*

Users are not allowed to add the same restaurant that already exists in the favorite list.

ExistFavorite

Ξ *FavPlace*
userId? : *USERID*
restId? : *RESTID*
response! : *RESPONSE*

---

restId? $\in$ *favPlace* $(\!|$ *favPlaceId* $(\!|\{userId?\}|\!)$ $|\!)$
response! = *existFavorite*

Users are not allowed to remove the restaurant that is not exists in the favorite list.

FavoriteNotExist

Ξ *FavPlace*
userId? : *USERID*
restId? : *RESTID*
response! : *RESPONSE*

---

restId? $\notin$ *favPlace* $(\!|$ *favPlaceId* $(\!|\{userId?\}|\!)$ $|\!)$
response! = *favoriteNotExist*

Only exisitng food orders can be rate by the user else the error will occurs.

OrderNotExist

Ξ *FoodOrder*
foodOrderId? : *FOODIRDERID*
userId? : *USERID*
response! : *RESPONSE*

---

foodOrderId? $\notin$ foodOrder $(\!|\{$*userId*?$\}|\!)$
response! = *orderNotExist*

Users can only rate the order with the valid number which is between 0 to 5. Otherwise, invalid rating will not be accepted by the system.

InvalidRating

rating? : $\mathbb{N}$
response! : *RESPONSE*

---

rating? $< 0 \lor$ *rating* > 5
response! = *invalidRating*

Users can only rate the order with the “to rate” state. Otherwise, invalid rating will not be accepted by the system.

StateNotToRate

Ξ *FoodOrder*
foodOrderId? : *FOODORDERID*
response! : *RESPONSE*

---

orderState $(\!|\{$*foodOrderId*?$\}|\!) \neq$ *to rate*
response! = *stateNotToRate*

8.5 Complete Schema

AddProductIntoBasketComplete ≙ (AddProductIntoBasket ∧ Success) ∨ UserNotExist ∨ UserNotLogin ∨ FoodNotExist ∨ InvalidQuantity

AddProductIntoBasketComplete

$userId, userId' : \mathbb{P}\ USERID$
$userName, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone, userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$basketId, basketId' : USERID \pfun FOODBASKETID$
$totalQty, totalQty' : FOODBASKETID \pfun TOTALFOODQTY$
$totalFoodPrice, totalFoodPrice' : FOODBASKETID \pfun TOTALFOODPRICE$

$itemId, itemId' : FOODBASKETID \pfun BASKETITEMID$
$basketFood, basketFood' : BASKETITEMID \pfun RESTFOODID$
$foodQty, foodQty' : BASKETITEMID \pfun FOODQTY$

$foodId, foodId' : RESTID \pfun RESTFOODID$
$foodName, foodName' : RESTFOODID \pfun RESTFOODNAME$
$foodPrice, foodPrice' : RESTFOODID \pfun RESTFOODPRICE$
$foodDes, foodDes' : RESTFOODID \pfun RESTFOODDES$

$restId, restId' : \mathbb{P}\ RESTID$
$restName, restName' : RESTID \pfun RESTNAME$
$restInfo, restInfo' : RESTID \pfun RESTINFO$
$restRate, restRate' : RESTID \pfun RESTRATE$

$userId? : USERID$
$itemId? : BASKETITEMID$
$foodId? : RESTFOODID$
$quantity? : \mathbb{N}$

$basket : FOODBASKETID$
$total : \mathbb{N}$
$price : \mathbb{N}$

$totalPrice : \mathbb{N}$

$response! : RESPONSE$

$((userId? \in userId \land userStatus(\!|\{userId?\}|\!) = loggedIn) \land userId? \in \{userId?\} \lhd basketId$
$\land FoodId? \in foodId \rhd \{foodId?\} \land quantity? \geqslant 1 \land basket = basketId (\!|\{userId?\}|\!) \land$
$total = \{basket\} \lhd totalQty \land total = total + quantity? \land price = \{foodId\} \lhd foodPrice \land$
$totalPrice = \{basket\} \lhd totalFoodPrice \land totalPrice = totalPrice + (price * quantity?) \land$
$itemId' = itemId \cup \{basketId \mapsto itemId?\} \land basketFood' = itemId \cup \{itemId? \mapsto foodId?\} \land$
$foodQty' = itemId \bigcup \{itemId? \mapsto quantity?\} \land basketId' = basketId \land totalQty' = totalQty \oplus \{basket \mapsto total\}$
$\land totalFoodPrice' = totalFoodPrice \oplus \{basket \mapsto totalPrice\} \land response! = success) \land$
$\text{dom } userName = \text{dom } userEmail = \text{dom } userGender = \text{dom } userPhone = \text{dom } userPassword =$
$userId \land \text{dom } userName' = \text{dom } userEmail' = \text{dom } userGender' = \text{dom } userPhone'$
$= \text{dom } userPassword' = userId' \land \text{dom } basketId = userId \land \text{dom } basketId' = userId' \land$
$foodQty \geqslant minFoodQty \land foodQty' \geqslant minFoodQty \land \text{dom } foodId = restId$
$\text{dom } foodName = \text{dom } foodPrice = \text{dom } foodDes = foodId \land \text{dom } foodId' = restId'$
$\text{dom } foodName' = \text{dom } foodPrice' = \text{dom } foodDes' = foodId'$
$\land \text{dom } restInfo = \text{dom } restRate = restId \land$
$\text{dom } restInfo' = \text{dom } restRate' = restId' \land minRate \leqslant restRate \geqslant maxRate \land$
$minRate \leqslant restRate' \geqslant maxRate)$
$\lor$
$($
$(userId? \notin userId \land response! = userNotExist \land$
$\text{dom } userName = \text{dom } userEmail = \text{dom } userGender = \text{dom } userPhone = \text{dom } userPassword =$
$userId \land \text{dom } userName' = \text{dom } userEmail' = \text{dom } userGender' = \text{dom } userPhone'$
$= \text{dom } userPassword' = userId')$
$\lor$
$(userStatus (\!|\{userId?\}|\!) \neq loggedIn \land response! = UserNotLogin \land$
$\text{dom } userName = \text{dom } userEmail = \text{dom } userGender = \text{dom } userPhone = \text{dom } userPassword =$
$userId \land \text{dom } userName' = \text{dom } userEmail' = \text{dom } userGender' = \text{dom } userPhone'$
$= \text{dom } userPassword' = userId')$
$\lor$
$(foodId? \notin foodId \rhd \{foodId?\} \land response! = foodNotExist \land \text{dom } foodId = restId \land$
$\text{dom } foodName = \text{dom } foodPrice = \text{dom } foodDes = foodId)$
$\lor$
$(quantity? < 0 \land response! = invalidQuantity)$
$\land$
$(userId' = userId \land userStatus' = userStatus \land foodId = foodId')$
$)$

ViewAllBasketProductComplete ≙ (ViewAllBasketProduct ∧ Success) ∨ UserNotExist ∨ UserNotLogin

ViewAllBasketProductComplete

$userId, userId' : \mathbb{P}\ USERID$
$userName, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone, userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$basketId, basketId' : USERID \pfun FOODBASKETID$
$totalQty, totalQty' : FOODBASKETID \pfun TOTALFOODQTY$
$totalFoodPrice, totalFoodPrice' : FOODBASKETID \pfun TOTALFOODPRICE$

$itemId, itemId' : FOODBASKETID \pfun BASKETITEMID$
$basketFood, basketFood' : BASKETITEMID \pfun RESTFOODID$
$foodQty, foodQty' : BASKETITEMID \pfun FOODQTY$

$foodId, foodId' : RESTID \pfun RESTFOODID$
$foodName, foodName' : RESTFOODID \pfun RESTFOODNAME$
$foodPrice, foodPrice' : RESTFOODID \pfun RESTFOODPRICE$
$foodDes, foodDes' : RESTFOODID \pfun RESTFOODDES$

$restId, restId' : \mathbb{P}\ RESTID$
$restName, restName' : RESTID \pfun RESTNAME$
$restInfo, restInfo' : RESTID \pfun RESTINFO$
$restRate, restRate' : RESTID \pfun RESTRATE$

$userId? : USERID$
$basketId? : FOODBASKETID$

$restId! : RESTFOODID$
$restName! : RESTNAME$
$basketItemId! : \mathbb{P}\ BASKETITEMID$
$basketFood! : \mathbb{P}\ RESTFOODID$
$basketfoodId : \mathbb{P}\ RESTFOODID$
$foodName! : \mathbb{P}\ RESTFOODNAME$
$foodPrice! : \mathbb{N}$

$foodQty! : \mathbb{N}$
$totalPrice! : \mathbb{N}$
$totalQty! : \mathbb{N}$

$response! : RESPONSE$

$((userId? \in userId \land userStatus(\!|\{userId?\}|\!) = loggedIn) \land userId? \in \{userId?\} \lhd basketId$
$\land basketId? \in \{userId?\} \lhd basketId \land basketItemId! = itemId\ (\!|\{basketId?\}|\!)$
$\land basketFood! = basketFood\ (\!|\{\ itemId\ (\!|\{basketId?\}|\!)\ \}|\!) \land basketfoodId = basketFood\ (\!|\{\ itemId\ (\!|\{basketId?\}|\!)\ \}|\!)$
$\land restId! = foodId \rhd \{\ basketfoodId\ \} \land restName! = restName\ (\!|\{\ foodId \rhd \{\ basketfoodId\ \}\}|\!)$
$\land foodName! = foodId\ (\!|\{\ basketfoodId\}|\!) \land foodPrice! = foodPrice\ (\!|\{\ basketfoodId\ \}|\!)$
$\land foodQty! = foodQty\ (\!|\{\ basketfoodId\ \}|\!) \land\ totalPrice! = totalFoodPrice\ (\!|\{basketId?\}|\!)$
$\land totalQty! = totalQty\ (\!|\{basketId?\}|\!) \land response! = success$
$\land \text{dom}\ userName = \text{dom}\ userEmail = \text{dom}\ userGender = \text{dom}\ userPhone = \text{dom}\ userPassword =$
$userId \land \text{dom}\ userName' = \text{dom}\ userEmail' = \text{dom}\ userGender' = \text{dom}\ userPhone'$
$= \text{dom}\ userPassword' = userId' \land \text{dom}\ basketId = userId \land \text{dom}\ basketId' = userId' \land$
$foodQty \geqslant minFoodQty \land foodQty' \geqslant minFoodQty \land \text{dom}\ foodId = restId$
$\text{dom}\ foodName = \text{dom}\ foodPrice = \text{dom}\ foodDes = foodId \land \text{dom}\ foodId' = restId'$
$\text{dom}\ foodName' = \text{dom}\ foodPrice' = \text{dom}\ foodDes' = foodId'$
$\land \text{dom}\ restInfo = \text{dom}\ restRate = restId \land$
$\text{dom}\ restInfo' = \text{dom}\ restRate' = restId' \land minRate \leqslant restRate \geqslant maxRate \land$
$minRate \leqslant restRate' \geqslant maxRate)$
$\lor$
$($
$(userId? \notin userId \land response! = userNotExist \land$
$\text{dom}\ userName = \text{dom}\ userEmail = \text{dom}\ userGender = \text{dom}\ userPhone = \text{dom}\ userPassword =$
$userId \land \text{dom}\ userName' = \text{dom}\ userEmail' = \text{dom}\ userGender' = \text{dom}\ userPhone'$
$= \text{dom}\ userPassword' = userId')$
$\lor$
$(userStatus\ (\!|\{userId?\}|\!) \neq loggedIn \land response! = userNotLogin \land$
$\text{dom}\ userName = \text{dom}\ userEmail = \text{dom}\ userGender = \text{dom}\ userPhone = \text{dom}\ userPassword =$
$userId \land \text{dom}\ userName' = \text{dom}\ userEmail' = \text{dom}\ userGender' = \text{dom}\ userPhone'$
$= \text{dom}\ userPassword' = userId')$
$\land$
$(userId' = userId \land userStatus' = userStatus)$
$)$

UpdateBasketProductComplete ≙ (UpdateBasketProduct ∧ Success) ∨ UserNotExist ∨ UserNotLogin ∨ ItemNotExist ∨ InvalidQuantity

UpdateBasketProductComplete

$userId, userId' : \mathbb{P}\ USERID$
$userName, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone, userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$basketId, basketId' : USERID \pfun FOODBASKETID$
$totalQty, totalQty' : FOODBASKETID \pfun TOTALFOODQTY$
$totalFoodPrice, totalFoodPrice' : FOODBASKETID \pfun TOTALFOODPRICE$

$itemId, itemId' : FOODBASKETID \pfun BASKETITEMID$
$basketFood, basketFood' : BASKETITEMID \pfun RESTFOODID$
$foodQty, foodQty' : BASKETITEMID \pfun FOODQTY$

$userId? : USERID$
$itemId? : ITEMID$
$quantity? : \mathbb{N}$

$basket : FOODBASKETID$
$totalprice : \mathbb{N}$
$price : N$
$totalqty : \mathbb{N}$
$qty : \mathbb{N}$

$response! : RESPONSE$

$((userId? \in userId \land userStatus(\!|\{userId?\}|\!) = loggedIn) \land userId? \in \{userId?\} \lhd basketId$
$\land basket = basketId\ (\!|\ \{userId?\}\ |\!) \land itemId? \in itemId\ (\!|\{basketId\}|\!) \land quantity \geqslant 1$
$\land itemId' = itemId \land basketFood = basketFood \land qty = foodQty\ (\!|\{itemId?\}|\!)$
$\land foodQty' = foodQty \oplus \{itemId? \mapsto quantity?\} \land totalqty = totalQty\ (\!|\ \{basket\}\ |\!)$
$\land totalprice = totalFoodPrice\ (\!|\ \{basket\}\ |\!) \land price = foodPrice\ (\!|\{\ \{itemId?\} \lhd basketFood\ \}|\!)$
$\land basketId' = basketId \land totalQty' = totalQty \oplus \{\ basket \mapsto (totalqty - qty + quantity?)\}$
$\land totalFoodPrice' = totalFoodPrice \oplus \{\ basket \mapsto (totalprice - (price*qty) + (price*quantity)\ \}$
$\land response! = success$

$\land$ dom *userName* = dom *userEmail* = dom *userGender* = dom *userPhone* = dom *userPassword* = *userId* $\land$ dom *userName'* = dom *userEmail'* = dom *userGender'* = dom *userPhone'* = dom *userPassword'* = *userId'* $\land$ dom *basketId* = *userId* $\land$ dom *basketId'* = *userId'* $\land$ *foodQty* $\geqslant$ *minFoodQty* $\land$ *foodQty'* $\geqslant$ *minFoodQty*)

$\lor$

(

(*userId*? $\notin$ *userId* $\land$ *response*! = *userNotExist* $\land$ dom *userName* = dom *userEmail* = dom *userGender* = dom *userPhone* = dom *userPassword* = *userId* $\land$ dom *userName'* = dom *userEmail'* = dom *userGender'* = dom *userPhone'* = dom *userPassword'* = *userId'*)

$\lor$

(*userStatus* $(\!|\{userId?\}|\!)$ $\neq$ *loggedIn* $\land$*response*! = *UserNotLogin* $\land$ dom *userName* = dom *userEmail* = dom *userGender* = dom *userPhone* = dom *userPassword* = *userId* $\land$ dom *userName'* = dom *userEmail'* = dom *userGender'* = dom *userPhone'* = dom *userPassword'* = *userId'*)

$\lor$

(*itemId*? $\notin$ *itemId* $(\!|\{ basketId\{userId?\} \}|\!)$ $\land$ *response* = *itemNotExist* $\land$ dom *basketId* = *userId* $\land$ *foodQty* $\geqslant$ *minFoodQty*)

$\lor$

(*quantity*? < 0 $\land$ *response*! = *invalidQuantity*)

$\land$

(*userId'* = *userId* $\land$ *userStatus'* = *userStatus* $\land$ *itemId* = *itemId'* $\land$ *basketId* = *basketId'*)

)

RemoveBasketProductComplete ≙ (RemoveBasketProduct ∧ Success) ∨ UserNotExist ∨ UserNotLogin ∨ ItemNotExist

RemoveBasketProductComplete

$userId, userId' : \mathbb{P}\ USERID$
$userName, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone, userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$basketId, basketId' : USERID \pfun FOODBASKETID$
$totalQty, totalQty' : FOODBASKETID \pfun TOTALFOODQTY$
$totalFoodPrice, totalFoodPrice' : FOODBASKETID \pfun TOTALFOODPRICE$

$itemId, itemId' : FOODBASKETID \pfun BASKETITEMID$
$basketFood, basketFood' : BASKETITEMID \pfun RESTFOODID$
$foodQty, foodQty' : BASKETITEMID \pfun FOODQTY$

$userId? : USERID$
$itemId? : BASKETITEMID$

$basket : FOODBASKETID$
$total : \mathbb{N}$
$qty : \mathbb{N}$
$totalPrice : \mathbb{N}$
$price : \mathbb{N}$

$response! = RESPONSE$

$((userId? \in userId \land userStatus(\{userId?\}) = loggedIn) \land userId? \in \{userId?\} \lhd basketId$
$\land basket = basketId\ (\{userId?\}) \land itemId \in itemId\ (\{basket\}) \land total = totalQty\ (\{basket\})$
$\land qty = foodQty\ (\{itemId?\}) \land totalPrice = totalFoodPrice\ (\{basket\})$
$\land price = foodPrice\ (basketFood\ (\{itemId?\})) \land itemId' = itemID \lhd \{itemId?\}$
$\land basketFood' = \{itemId?\} \lhd basketFood \land foodQty' = \{itemId?\} \lhd foodQty$
$\land basketId' = basketId \land totalQty' = totalQty \oplus \{basket \mapsto (total - qty)\}$
$\land totalFoodPrice' = totalFoodPrice \oplus \{basket \mapsto (totalPrice - (price*qty))\}$
$\land response! = success \land$
dom *userName* = dom *userEmail* = dom *userGender* = dom *userPhone* = dom *userPassword* =

$userId \land \text{dom}\ userName' = \text{dom}\ userEmail' = \text{dom}\ userGender' = \text{dom}\ userPhone'$
$= \text{dom}\ userPassword' = userId' \land \text{dom}\ basketId = userId \land \text{dom}\ basketId' = userId' \land$
$foodQty \geqslant minFoodQty \land foodQty' \geqslant minFoodQty)$
$\lor$

(
$(userId? \notin userId \land response! = userNotExist \land$
$\text{dom}\ userName = \text{dom}\ userEmail = \text{dom}\ userGender = \text{dom}\ userPhone = \text{dom}\ userPassword =$
$userId \land \text{dom}\ userName' = \text{dom}\ userEmail' = \text{dom}\ userGender' = \text{dom}\ userPhone'$
$= \text{dom}\ userPassword' = userId')$
$\lor$
$(userStatus\ (\!|\{userId?\}|\!) \neq loggedIn \land response! = UserNotLogin \land$
$\text{dom}\ userName = \text{dom}\ userEmail = \text{dom}\ userGender = \text{dom}\ userPhone = \text{dom}\ userPassword =$
$userId \land \text{dom}\ userName' = \text{dom}\ userEmail' = \text{dom}\ userGender' = \text{dom}\ userPhone'$
$= \text{dom}\ userPassword' = userId')$
$\lor$
$(itemId? \notin itemId\ (\!|\{\ basketId\{userId?\}\ \}|\!) \land response = itemNotExist \land foodQty \geqslant minFoodQty$
$\land \text{dom}\ basketId = userId)$
$\land$
$(userId' = userId \land userStatus' = userStatus \land itemId = itemId?)$
)

AddFavoriteComplete ≙ (AddFavoriteBasket ∧ Success) ∨ UserNotExist ∨ UserNotLogin ∨ RestuarantNotExist ∨ MaxFavoriteReached ∨ ExistFavorite

AddFavoriteComplete

$userId, userId' : \mathbb{P}\ USERID$
$userName, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone, userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$restId, restId' : \mathbb{P}\ RESTID$
$restName, restName' : RESTID \pfun RESTNAME$
$restInfo, restInfo' : RESTID \pfun RESTINFO$
$restRate, restRate' : RESTID \pfun RESTRATE$

$favPlaceId, favPlaceId' : USERID \pfun FAVPLACEID$
$favPlace, favPlace' : FAVPLACEID \pfun RESTID$

$userId? : USERID$
$favId : FAVPLACEID$
$restId? : RESTID$

$response! = RESPONSE$

$((userId? \in userId \land userStatus(\!|\{userId?\}|\!) = loggedIn) \land restId? \in restId$
$\land \#\{favPlaceId\ (\!|\{userId?\}|\!)\} < 100 \land favId = favPlaceId\ (\!|\{userId?\}|\!)$
$\land restId? \notin favPlace\ (\!|\{favId\}|\!) \land favPlace' = favPlace \cup \{favId \mapsto restId?\}$
$\land response! = success \land$
$\text{dom}\ userName = \text{dom}\ userEmail = \text{dom}\ userGender = \text{dom}\ userPhone = \text{dom}\ userPassword =$
$userId \land \text{dom}\ userName' = \text{dom}\ userEmail' = \text{dom}\ userGender' = \text{dom}\ userPhone'$
$= \text{dom}\ userPassword' = userId' \land \text{dom}\ basketId = userId \land \text{dom}\ basketId' = userId' \land$
$\land \text{dom}\ restInfo = \text{dom}\ restRate = restId \land$
$\text{dom}\ restInfo' = \text{dom}\ restRate' = restId' \land minRate \leqslant restRate \geqslant maxRate \land$
$minRate \leqslant restRate' \geqslant maxRate \land \text{dom}\ favPlaceId = userId \land \text{dom}\ favPlaceId' = userId')$
$\lor$
$($
$(userId? \notin userId \land response! = userNotExist \land$
$\text{dom}\ userName = \text{dom}\ userEmail = \text{dom}\ userGender = \text{dom}\ userPhone = \text{dom}\ userPassword =$

$$
\begin{array}{l}
userId \land \text{dom}\ userName' = \text{dom}\ userEmail' = \text{dom}\ userGender' = \text{dom}\ userPhone' \\
= \text{dom}\ userPassword' = userId') \\
\lor \\
(userStatus\ (\!|\{userId?\}|\!) \neq loggedIn \land response! = UserNotLogin \land \\
\text{dom}\ userName = \text{dom}\ userEmail = \text{dom}\ userGender = \text{dom}\ userPhone = \text{dom}\ userPassword = \\
userId \land \text{dom}\ userName' = \text{dom}\ userEmail' = \text{dom}\ userGender' = \text{dom}\ userPhone' \\
= \text{dom}\ userPassword' = userId') \\
\lor \\
(restId? \notin restId \land response! = restaurantNotExist \land \\
\text{dom}\ restInfo = \text{dom}\ restRate = restId \land minRate \leqslant restRate \geqslant maxRate) \\
\lor \\
(\#(favPlaceId\ (\!|\{userId?\}|\!)) \geqslant 100 \land response! = maxFavoriteReached \land \text{dom}\ favPlaceId = userId) \\
\lor \\
(restId? \in favPlace\ (\!|\ favPlaceId\ (\!|\{userId?\}|\!)\ |\!) \land response! = existFavorite \land \text{dom}\ favPlaceId = userId) \\
\land \\
(userId' = userId \land userStatus' = userStatus \land restId' = restId \land favPlaceId = favPlaceId' \\
\land favPlace' = favPlace) \\
)
\end{array}
$$

ViewAllFavoriteComplete ≙ (ViewAllFavorite ∧ Success) ∨ UserNotExist ∨ UserNotLogin

ViewAllFavoriteComplete

$userId, userId' : \mathbb{P}\ USERID$
$userName, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone, userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$restId, restId' : \mathbb{P}\ RESTID$
$restName, restName' : RESTID \pfun RESTNAME$
$restInfo, restInfo' : RESTID \pfun RESTINFO$
$restRate, restRate' : RESTID \pfun RESTRATE$

$userId? : USERID$

$favPlaceId! : FAVPLACEID$
$rest : \mathbb{P}\ RESTID$
$restName! : \mathbb{P}\ RESTNAME$
$restRate! : \mathbb{P}\ RESTRATE$

$response! = RESPONSE$

$((userId? \in userId \land userStatus(\!|\{userId?\}|\!) = loggedIn) \land favPlaceId! = favPlaceId\ (\!|\{userId?\}|\!)$
$\land\ rest = favPlace\ (\!|\{\ favPlaceId\ (\!|\{userId?\}|\!)\ \}|\!) \land restName! = restName\ (\!|\{rest\}|\!)$
$\land\ restRate! = restRate\ (\!|\{rest\}|\!) \land response! = success$
$\land\ \mathrm{dom}\ userName = \mathrm{dom}\ userEmail = \mathrm{dom}\ userGender = \mathrm{dom}\ userPhone = \mathrm{dom}\ userPassword =$
$userId \land \mathrm{dom}\ userName' = \mathrm{dom}\ userEmail' = \mathrm{dom}\ userGender' = \mathrm{dom}\ userPhone'$
$= \mathrm{dom}\ userPassword' = userId' \land \mathrm{dom}\ basketId = userId \land \mathrm{dom}\ basketId' = userId' \land$
$\land\ \mathrm{dom}\ restInfo = \mathrm{dom}\ restRate = restId \land$
$\mathrm{dom}\ restInfo' = \mathrm{dom}\ restRate' = restId' \land minRate \leqslant restRate \geqslant maxRate \land$
$minRate \leqslant restRate' \geqslant maxRate \land \mathrm{dom}\ favPlaceId = userId \land \mathrm{dom}\ favPlaceId' = userId')$
$\lor$
$($
$(userId? \notin userId \land response! = userNotExist \land$
$\mathrm{dom}\ userName = \mathrm{dom}\ userEmail = \mathrm{dom}\ userGender = \mathrm{dom}\ userPhone = \mathrm{dom}\ userPassword =$
$userId \land \mathrm{dom}\ userName' = \mathrm{dom}\ userEmail' = \mathrm{dom}\ userGender' = \mathrm{dom}\ userPhone'$
$= \mathrm{dom}\ userPassword' = userId')$
$\lor$

(*userStatus* ⦇{*userId*?}⦈ ≠ *loggedIn* ∧*response*! = *UserNotLogin* ∧
dom *userName* = dom *userEmail* = dom *userGender* = dom *userPhone* = dom *userPassword* =
userId ∧ dom *userName'* = dom *userEmail'* = dom *userGender'* = dom *userPhone'*
= dom *userPassword'* = *userId'*)
∧
(*userId'* = *userId* ∧ *userStatus'* = *userStatus*)
)

RemoveFavoriteComplete ≙ (RemoveFavorite ∧ Success) ∨ UserNotExist ∨ UserNotLogin ∨ RestuarantNotExist ∨ FavoriteNotExist

RemoveFavoriteComplete

$userId, userId' : \mathbb{P}\ USERID$
$userName, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone, userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$restId, restId' : \mathbb{P}\ RESTID$
$restName, restName' : RESTID \pfun RESTNAME$
$restInfo, restInfo' : RESTID \pfun RESTINFO$
$restRate, restRate' : RESTID \pfun RESTRATE$

$userId? : USERID$
$favId : FAVPLACEID$
$restId? : RESTID$

$response! = RESPONSE$

$((userId? \in userId \land userStatus(\!|\{userId?\}|\!) = loggedIn) \land favId \in favPlaceId\ (\!|\{userId?\}|\!)$
$\land restId? \in restId \land restId? \in favPlace\ (\!|\{favId\}|\!)$
$\land favPlace' = favPlace \setminus \{favId? \mapsto restId?\} \land response! = success \land$
$\mathrm{dom}\ userName = \mathrm{dom}\ userEmail = \mathrm{dom}\ userGender = \mathrm{dom}\ userPhone = \mathrm{dom}\ userPassword =$
$userId \land \mathrm{dom}\ userName' = \mathrm{dom}\ userEmail' = \mathrm{dom}\ userGender' = \mathrm{dom}\ userPhone'$
$= \mathrm{dom}\ userPassword' = userId' \land \mathrm{dom}\ basketId = userId \land \mathrm{dom}\ basketId' = userId' \land$
$\land \mathrm{dom}\ restInfo = \mathrm{dom}\ restRate = restId \land$
$\mathrm{dom}\ restInfo' = \mathrm{dom}\ restRate' = restId' \land minRate \leqslant restRate \geqslant maxRate \land$
$minRate \leqslant restRate' \geqslant maxRate \land \mathrm{dom}\ favPlaceId = userId \land \mathrm{dom}\ favPlaceId' = userId')$
$\lor$
$($
$(userId? \notin userId \land response! = userNotExist \land$
$\mathrm{dom}\ userName = \mathrm{dom}\ userEmail = \mathrm{dom}\ userGender = \mathrm{dom}\ userPhone = \mathrm{dom}\ userPassword =$
$userId \land \mathrm{dom}\ userName' = \mathrm{dom}\ userEmail' = \mathrm{dom}\ userGender' = \mathrm{dom}\ userPhone'$
$= \mathrm{dom}\ userPassword' = userId')$
$\lor$
$(userStatus\ (\!|\{userId?\}|\!) \neq loggedIn \land response! = UserNotLogin \land$

dom *userName* = dom *userEmail* = dom *userGender* = dom *userPhone* = dom *userPassword* = *userId* ∧ dom *userName'* = dom *userEmail'* = dom *userGender'* = dom *userPhone'* = dom *userPassword'* = *userId'*)

∨

(*restId*? ∉ *restId* ∧ *response*! = *restaurantNotExist* ∧ dom *restInfo* = dom *restRate* = *restId* ∧ *minRate* ⩽ *restRate* ⩾ *maxRate*)

∨

(*restId*? ∉ *favPlace* (⦇ *favPlaceId* (⦇{*userId*?}⦈) ⦈ ∧ *response*! = *favoriteNotExist* ∧ dom *favPlaceId* = *userId*)

∧

(*userId'* = *userId* ∧ *userStatus'* = *userStatus* ∧ *restId* = *restId'* ∧ *favPlace* = *favPlace'* ∧ *favPlaceId* = *favPlaceId'*)

)

ViewAllOrderComplete ≙ (ViewAllOrder ∧ Success) ∨ UserNotExist ∨ UserNotLogin

ViewAllOrderComplete

$userId, userId' : \mathbb{P}\ USERID$
$userName, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone, userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$basketId, basketId' : USERID \rightarrowtail FOODBASKETID$
$totalQty, totalQty' : FOODBASKETID \rightarrowtail TOTALFOODQTY$
$totalFoodPrice, totalFoodPrice' : FOODBASKETID \rightarrowtail TOTALFOODPRICE$

$foodOrderId, foodOrderId' : USERID \rightarrowtail FOODORDERID$
$orderBasket, orderBasket' : FOODORDERID \rightarrowtail BASKETID$
$orderState, orderState' : FOODORDERID \rightarrowtail ORDERSTATE$
$orderDate, orderDate' : FOODORDERID \rightarrowtail ORDERDATE$
$orderPayment, orderPayment' : FOODORDERID \rightarrowtail ORDERPAYMENT$
$orderRate, orderRate : FOODORDERID \rightarrowtail ORDERRATE$

$userId? : USERID$

$foodOrderId! : \mathbb{P}\ FOODORDERID$
$order : \mathbb{P}\ FOODORDERID$
$basketId : \mathbb{P}\ FOODBASKETID$
$totalQty! : \mathbb{P}\ TOTALFOODQTY$
$totalFoodPrice! : \mathbb{P}\ TOTALFOODPRICE$
$state! : \mathbb{P}\ ORDERSTATE$
$payment! : \mathbb{P}\ ORDERPAYMENT$
$date! : \mathbb{P}\ ORDERDATE$

$response! = RESPONSE$

$((userId? \in userId \land userStatus(\!|\{userId?\}|\!) = loggedIn) \land order = foodOrderId\ (\!|\{userId?\}|\!)$
$\land basketId = basketId\ (\!|\{order\}|\!) \land foodOrderId! = order \land totalQty! = totalQty\ (\!|\{basketId\}|\!)$
$\land totalFoodPrice! = totalFoodPrice\ (\!|\{basketId\}|\!) \land state! = orderState\ (\!|\{order\}|\!)$
$\land payment! = orderPayment\ (\!|\{order\}|\!) \land date! = orderDate\ (\!|\{order\}|\!)$
$\land response! = success \land$
$\text{dom}\ userName = \text{dom}\ userEmail = \text{dom}\ userGender = \text{dom}\ userPhone = \text{dom}\ userPassword =$
$userId \land \text{dom}\ userName' = \text{dom}\ userEmail' = \text{dom}\ userGender' = \text{dom}\ userPhone'$

= dom *userPassword'* = *userId'* ∧ dom *basketId* = *userId* ∧ dom *basketId'* = *userId'* ∧
dom *foodOrderId* = *userId* ∧ dom *foodOrderId'* = *userId'* ∧ ran *orderBasket* = *basketId* ∧
ran *orderBasket'* = *basketId'*)
∨
(
(*userId*? ∉ *userId* ∧ *response*! = *userNotExist* ∧
dom *userName* = dom *userEmail* = dom *userGender* = dom *userPhone* = dom *userPassword* =
userId ∧ dom *userName'* = dom *userEmail'* = dom *userGender'* = dom *userPhone'*
= dom *userPassword'* = *userId'*)
∨
(*userStatus* (|{*userId*?}|) ≠ *loggedIn* ∧*response*! = *UserNotLogin* ∧
dom *userName* = dom *userEmail* = dom *userGender* = dom *userPhone* = dom *userPassword* =
userId ∧ dom *userName'* = dom *userEmail'* = dom *userGender'* = dom *userPhone'*
= dom *userPassword'* = *userId'*)
∧
(*userId'* = *userId* ∧ *userStatus'* = *userStatus*)
)

ViewOrderWithStateComplete ≙ (ViewOrderWithState ∧ Success) ∨ UserNotExist ∨ UserNotLogin ∨ InvalidState

ViewOrderWithStateComplete

$userId, userId' : \mathbb{P}\ USERID$
$userName, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone, userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$basketId, basketId' : USERID \pfun FOODBASKETID$
$totalQty, totalQty' : FOODBASKETID \pfun TOTALFOODQTY$
$totalFoodPrice, totalFoodPrice' : FOODBASKETID \pfun TOTALFOODPRICE$

$foodOrderId, foodOrderId' : USERID \pfun FOODORDERID$
$orderBasket, orderBasket' : FOODORDERID \pfun BASKETID$
$orderState, orderState' : FOODORDERID \pfun ORDERSTATE$
$orderDate, orderDate' : FOODORDERID \pfun ORDERDATE$
$orderPayment, orderPayment' : FOODORDERID \pfun ORDERPAYMENT$
$orderRate, orderRate : FOODORDERID \pfun ORDERRATE$

$foodOrderId! : \mathbb{P}\ FOODORDERID$
$order : \mathbb{P}\ FOODORDERID$
$basketId : \mathbb{P}\ FOODBASKETID$
$totalQty! : \mathbb{P}\ TOTALFOODQTY$
$totalFoodPrice! : \mathbb{P}\ TOTALFOODPRICE$
$state! : \mathbb{P}\ ORDERSTATE$
$payment! : \mathbb{P}\ ORDERPAYMENT$
$date! : \mathbb{P}\ ORDERDATE$

$response! = RESPONSE$

$((userId? \in userId \land userStatus(\!|\{userId?\}|\!) = loggedIn) \land$
$order = foodOrderId\ (\!|\{userId?\}|\!) \land \{foodOrderId\ (\!|\{userId?\}|\!)\} \lhd orderStatus = status? \land$
$basketId = basketId\ (\!|\{order\}|\!) \land foodOrderId! = order \land totalQty! = totalQty\ (\!|\{basketId\}|\!) \land$
$totalFoodPrice! = totalFoodPrice\ (\!|\{basketId\}|\!) \land state! = orderState\ (\!|\{order\}|\!) \land$
$payment! = orderPayment\ (\!|\{order\}|\!) \land date! = orderDate\ (\!|\{order\}|\!) \land$
$response! = success \land$

dom *userName* = dom *userEmail* = dom *userGender* = dom *userPhone* = dom *userPassword* = *userId* ∧ dom *userName'* = dom *userEmail'* = dom *userGender'* = dom *userPhone'* = dom *userPassword'* = *userId'* ∧ dom *basketId* = *userId* ∧ dom *basketId'* = *userId'* ∧ dom *foodOrderId* = *userId* ∧ dom *foodOrderId'* = *userId'* ∧ ran *orderBasket* = *basketId* ∧ ran *orderBasket'* = *basketId'*)

∨

(

(*userId*? ∉ *userId* ∧ *response*! = *userNotExist* ∧ dom *userName* = dom *userEmail* = dom *userGender* = dom *userPhone* = dom *userPassword* = *userId* ∧ dom *userName'* = dom *userEmail'* = dom *userGender'* = dom *userPhone'* = dom *userPassword'* = *userId'*)

∨

(*userStatus* (|{*userId*?}|) ≠ *loggedIn* ∧*response*! = *UserNotLogin* ∧ dom *userName* = dom *userEmail* = dom *userGender* = dom *userPhone* = dom *userPassword* = *userId* ∧ dom *userName'* = dom *userEmail'* = dom *userGender'* = dom *userPhone'* = dom *userPassword'* = *userId'*)

∧

(*userId'* = *userId* ∧ *userStatus'* = *userStatus*)

)

RateOrderComplete ≙ (RateOrder ∧ Success) ∨ UserNotExist ∨ UserNotLogin ∨ OrderNotExist ∨ RestuarantNotExist ∨ InvalidRating ∨ stateNotToRate

RateOrderComplete

$userId, userId' : \mathbb{P}\ USERID$
$userName, userName' : \mathbb{P}\ (USERID \times USERNAME)$
$userGender, userGender' : \mathbb{P}\ (USERID \times USERGENDER)$
$userPhone, userPhone' : \mathbb{P}\ (USERID \times USERPHONE)$
$userPassword, userPassword' : \mathbb{P}\ (USERID \times PASSWORD)$
$userHistory, userHistory' : \mathbb{P}\ (USERID \times USERHISTORY)ad$
$userStatus, userStatus' : \mathbb{P}\ (USERID \times USERSTATUS)$

$basketId, basketId' : USERID \rightarrowtail FOODBASKETID$
$totalQty, totalQty' : FOODBASKETID \rightarrowtail TOTALFOODQTY$
$totalFoodPrice, totalFoodPrice' : FOODBASKETID \rightarrowtail TOTALFOODPRICE$

$foodOrderId, foodOrderId' : USERID \rightarrowtail FOODORDERID$
$orderBasket, orderBasket' : FOODORDERID \rightarrowtail BASKETID$
$orderState, orderState' : FOODORDERID \rightarrowtail ORDERSTATE$
$orderDate, orderDate' : FOODORDERID \rightarrowtail ORDERDATE$
$orderPayment, orderPayment' : FOODORDERID \rightarrowtail ORDERPAYMENT$
$orderRate, orderRate : FOODORDERID \rightarrowtail ORDERRATE$

$restId, restId' : \mathbb{P}\ RESTID$
$restName, restName' : RESTID \rightarrowtail RESTNAME$
$restInfo, restInfo' : RESTID \rightarrowtail RESTINFO$
$restRate, restRate' : RESTID \rightarrowtail RESTRATE$

$userId? : USERID$
$foodOrderId? : FOODORDERID$
$restId? : RESTID$
$rating? : \mathbb{N}$

$response! = RESPONSE$

$((userId? \in userId \land userStatus(\!|\{userId?\}|\!) = loggedIn) \land$
$foodOrderId? \in foodOrderId\ (\!|\{userId?\}|\!) \land foodState\ (\!|\{foodOrderId?\}|\!) = to\ rate$
$\land\ restId? \in restId \land rating = rating? \geqslant 0 \land rating? \leqslant 5$
$\land\ basketId' = basketId \land totalQty' = totalQty$
$\land\ totalFoodPrice' = totalFoodPrice \land foodOrderId' = foodOrderId$
$\land\ orderBasket' = orderBasket \land orderState' = orderStates \oplus \{foodOrderId \mapsto complete\}$

$$
\begin{array}{l}
\land\ orderDate' = orderDate \land orderPayment' = orderPayment \\
\land\ orderRate' = orderRate \oplus \{foodOrderId \mapsto rating?\} \land orderDate' = orderDate \\
\land\ response! = success \land \\
\mathrm{dom}\ userName = \mathrm{dom}\ userEmail = \mathrm{dom}\ userGender = \mathrm{dom}\ userPhone = \mathrm{dom}\ userPassword = \\
userId \land \mathrm{dom}\ userName' = \mathrm{dom}\ userEmail' = \mathrm{dom}\ userGender' = \mathrm{dom}\ userPhone' \\
= \mathrm{dom}\ userPassword' = userId' \land \mathrm{dom}\ basketId = userId \land \mathrm{dom}\ basketId' = userId' \land \\
\mathrm{dom}\ foodOrderId = userId \land \mathrm{dom}\ foodOrderId' = userId' \land \mathrm{ran}\ orderBasket = basketId \land \\
\mathrm{ran}\ orderBasket' = basketId' \land \mathrm{dom}\ restInfo = \mathrm{dom}\ restRate = restId \land \\
\mathrm{dom}\ restInfo' = \mathrm{dom}\ restRate' = restId' \land minRate \leqslant restRate \geqslant maxRate \\
\land\ minRate \leqslant restRate' \geqslant maxRate) \\
\lor \\
(\\
(userId? \notin userId \land response! = userNotExist \land \\
\mathrm{dom}\ userName = \mathrm{dom}\ userEmail = \mathrm{dom}\ userGender = \mathrm{dom}\ userPhone = \mathrm{dom}\ userPassword = \\
userId \land \mathrm{dom}\ userName' = \mathrm{dom}\ userEmail' = \mathrm{dom}\ userGender' = \mathrm{dom}\ userPhone' \\
= \mathrm{dom}\ userPassword' = userId') \\
\lor \\
(userStatus\ (\!|\{userId?\}|\!) \neq loggedIn \land response! = UserNotLogin \land \\
\mathrm{dom}\ userName = \mathrm{dom}\ userEmail = \mathrm{dom}\ userGender = \mathrm{dom}\ userPhone = \mathrm{dom}\ userPassword = \\
userId \land \mathrm{dom}\ userName' = \mathrm{dom}\ userEmail' = \mathrm{dom}\ userGender' = \mathrm{dom}\ userPhone' \\
= \mathrm{dom}\ userPassword' = userId') \\
\lor \\
(foodOrderId? \notin \mathrm{foodOrder}\ (\!|\{userId?\}|\!) \land response! = orderNotExist \land \\
\mathrm{dom}\ foodOrderId' = userId' \land \mathrm{ran}\ orderBasket = basketId) \\
\lor \\
(restId? \notin restId \land response! = restaurantNotExist \\
\land\ \mathrm{dom}\ restInfo = \mathrm{dom}\ restRate = restId \land \\
\mathrm{dom}\ restInfo' = \mathrm{dom}\ restRate' = restId' \land minRate \leqslant restRate \geqslant maxRate \\
\land\ minRate \leqslant restRate' \geqslant maxRate) \\
\lor \\
(orderState\ (\!|\{foodOrderId?\}|\!) \neq to\ rate \land response! = stateNotToRate \land \\
\mathrm{dom}\ foodOrderId' = userId' \land \mathrm{ran}\ orderBasket = basketId) \\
\land \\
(userId' = userId \land userStatus' = userStatus \land foodOrder' = foodOrder \land orderState' = orderState) \\
)
\end{array}
$$

8.6 Conclusion

In this Formal Methods assignment, I gained valuable insights into the practical application of Z schemas within the context of a real-world system, Shopee. This new methodology has significantly contributed to enhancing software quality by enabling a comprehensive description of system behavior and properties, all without directly modifying the underlying code. I am sincerely grateful to my dedicated lecturer, Ms. Azuwati, for providing the necessary courses and materials that laid the foundation for my understanding of formal methods. Her patient and accessible guidance greatly enriched my learning experience. I also want to express my appreciation for the invaluable guidance and support from my tutor, Ms. Mazlinda. Her patience and assistance were instrumental in addressing my concerns and clarifying any confusion I had throughout the assignment. Without her guidance, I might not have been able to complete the assignment correctly. Finally, I extend my heartfelt thanks to my teammates, who collaborated tirelessly with me throughout this assignment. Their teamwork and dedication significantly contributed to our success.