Dish Ordering System

Simon Wang

Table of Contents

1. Project scope and project description

Our project is going to build a Dish Ordering System (DOS) which help the customer to order the dish. Also, DOS provides the function for staffs to view dish inventory which helps them serve the customer better. DOS will display the dishes information on the screen, the customers will select the desired dish and add it to the cart. Once the dish is finalized, customers can place the order through this system. DOS provides admin account for the staff. Staffs can view and modify all of the placed order, including the dishes name, price, inventory etc. DOS will calculate the subtotal and total price for each order and provide the payment method for customers. For iteration 1, DOS provides the function mentions above, in the following iteration, more function will be added.

2. Functional requirements of Dish Ordering System

R1. The system must allow a customer to view the dish name, price, and the availability.

R2. The system must check the inventory and show the availability of all dishes to the customer.

R3. The system must allow the customer to add dishes into his order and leave comments.

R4. The system must collect order information generated by the customer and display the order details to the customer.

R5. The system must allow the customer to place an order.

R6. The system must show all placed orders to the staff.

R7. The system must automatically update the inventory of each dish.

R8. The system must allow the staff to login to the system as an administrator and logout.

R9. The system must allow the administrator to modify the dish description, including name, price, and inventory.

R10. The system must allow the administrator to cancel an order placed by the customer.

R11. The system must be able to automatically update the subtotal and total of an order when the customer updates the order.

R12. The system must allow a customer to select a payment method to make a payment.

R13. The system must provide a storage to store dishes, orders and staff login information.

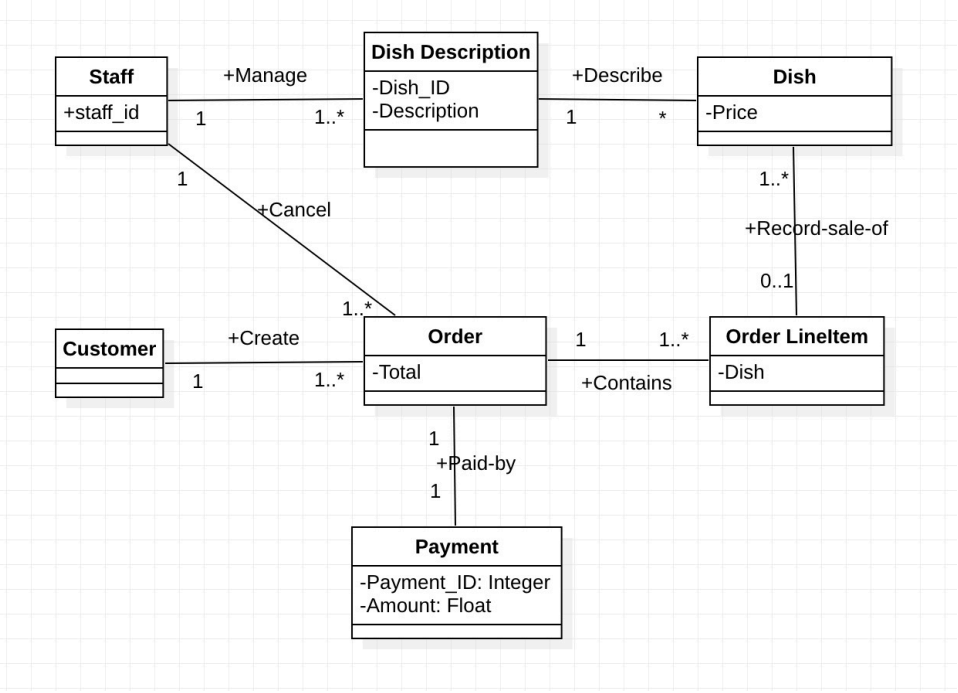
3. Plan for each iteration

Iteration 1		**Sep. 10 – Oct. 1**
Requirements Gathering	All group 9 members	Sep. 8
Planning for each iteration	All group 9 members	Sep. 8
Environment setup	All group 9 members	Sep. 8
Domain model design	All group 9 members	Sep. 15
Use Case Model Design	All group 9 members	Sep. 15 – Sep. 22
Class diagram design	All group 9 members	Sep. 22 – Sep. 29
Coding, testing, integration (Finishing 2 use cases)	All group 9 members	Sep. 29 – Oct. 1
Submitting report	X	Oct. 1
Demo & feedbacks collection	All group 9 members	Oct. 1

Iteration 2		**Oct. 2 – Oct. 30**
Revise project report based on feedbacks	Hao-Lun Lo	Oct. 2 – Oct. 6
Database design and setup	Song Ao, Chongliang He	Oct. 6
Coding, testing, (Finishing all use cases and user interface)	All group 9 members	Oct. 6 – Oct. 13
Sequence diagram design	All group 9 members	Oct. 13
Adding requirements	YuChuan Lin, Feiyang Zhao	Oct. 13
UC, SD, CD revision	You Jung Chiang, Shih-Han Wang	Oct. 20 – Oct. 27
Coding, testing, integration	All group 9 members	Oct. 20 – Oct. 27
Submitting report	X	Oct. 29
Demo & feedbacks collection	All group 9 members	Oct. 31

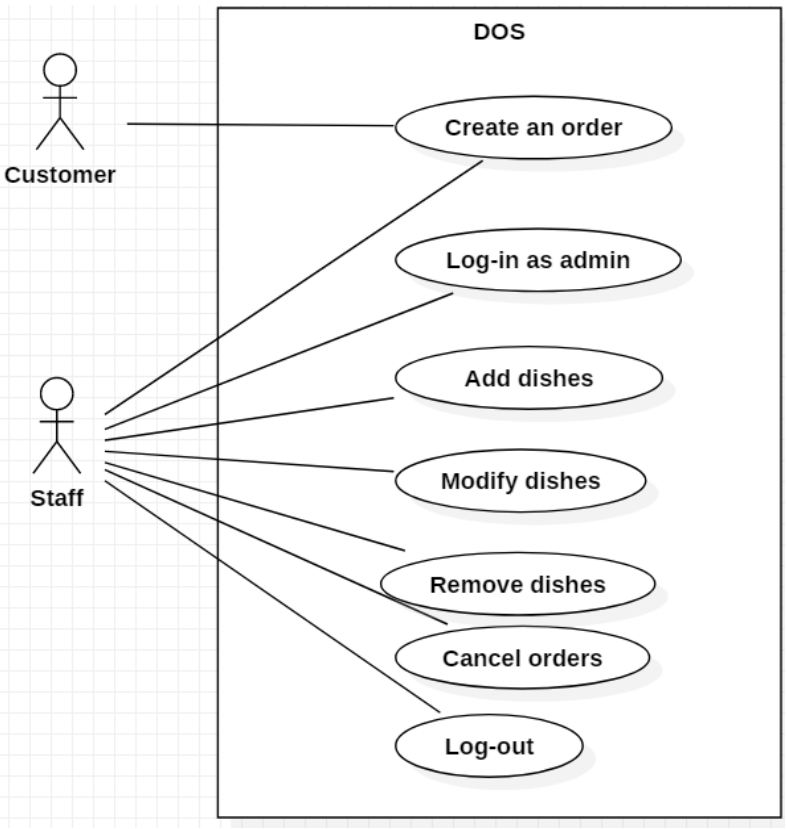
Iteration 3		**Nov. 1 – Nov. 28**
Revise project report based on feedbacks	Hao-Lun Lo	Nov. 1 – Nov. 3
Class diagram revision (Applying design pattern)	All group 9 members	Nov.3 – Nov. 28
Coding & integration	All group 9 members	Nov. 10 – Nov. 16
Testing with JUnit	All group 9 members	Nov. 10 – Nov. 16
Submitting report	X	Nov. 28
Final demonstration	All group 9 members	Dec. 3

4. Domain Model



5. Use Case

5.1 Use case diagram



5.2 Fully Dressed Use Cases

1) Use Case 1

Name: Create an order

Scope: Dish Ordering System (DOS)

Level: User-goal

Actors: Staff (Primary), DOS (Supporting)

Stakeholders:

Customer: Get desired dishes.

Staff: Collect payments from customers.

Preconditions: Dish Ordering System is ready for use.

Main Success Scenario:

1. TUCBW: the customer taps on the “menu” button.
2. The system displays the menu and an empty order.
3. The customer selects desired dishes.
4. The system displays selected dishes, including subtotal, in the order.
5. The customer taps on the check-out button.
6. The system shows the total and waits for the customer to swipe card.
7. The customer swipes his payment card.

8. The system processes the payment, save the order and returns a payment success message.
9. TUCEW: the customer sees the success message.

Alternative Flows:

3a. The selected dish has been sold out:

1. The system shows a notice message.

3b. The customer taps the “Checkout” button by accident:

1. The system shows a “Back To Cart” button.
2. The customer continues adding dishes into order.

Post-condition (Success Guarantee): An order has been created.

2) Use Case 2

Name: Log-in as admin

Scope: Dish Ordering System (DOS)

Level: User-goal level

Actors: Staff (Primary), DOS (Supporting)

Stakeholders: Staff: Login to DOS as an administrator to modify menu and orders.

Preconditions: The staff id and the password has been preset; the staff knows his staff id and password to login to DOS.

Main Success Scenario:

1. TUCBW: the staff taps “Admin-Mode” button.
2. The system displays the login page.
3. The staff enters his staff id and password and then taps “Login” button.
4. The system checks if the staff id and the password match the user information in the system, then it shows the admin-only page.
5. TUCEW: the staff has logged in successfully.

Alternative Flows:

3.a The staff enters incorrect staff id or password:

1. The system shows a “Wrong Password” message.
2. The system redirects to the login page.

3.b The customer clicks on the admin-mode by accident:

1. The system shows a “Return” in the login page.
2. The customer clicks on the button.
3. The system redirects to the home page.

Post-Conditions (Success Guarantee): The staff has logged in to DOS and is able to modify orders and menu.

3) Use Case 3

Name: Add dishes

Scope: Dish Ordering System (DOS)

Level: User-goal level

Actors: Staff (Primary), DOS (supporting)

Stakeholders: Staff: adds new dishes into the menu, including their name, price and inventory.

Preconditions: Staff has logged in to the system.

Main Success Scenario:

1. TUCBW: the staff chooses to add dishes in the admin mode.
2. The system displays a form with three fields: name, price and inventory, for adding a dish.
3. The staff fills up the form and submit.
4. The system adds a new dish and redirects to “modify dish” page.
5. TUCEW: the staff sees the new dish in the modify dish page.

Alternative Flows:

3.a The staff adds a dish with duplicated name that exists in the system already:

1. The system shows an error message to the staff and redirects to the “modify dish” page.
2. The staff will not see duplicated dishes.

Post-Conditions (Success Guarantee): A new dish has been added to the menu.

4) Use Case 4

Name: Modify dishes

Scope: Dish Ordering System (DOS)

Level: User-goal level

Actors: Staff (Primary), DOS (supporting)

Stakeholders: Staff: Modify dishes in the menu, including changing name, price, and quantity.

Preconditions: Staff has logged in to the system.

Main Success Scenario:

1. TUCBW: The use case begins when the staff chooses to modify dishes.
2. The system displays the menu to staff.
3. The staff modifies a dish by entering its name, price or inventory then clicks on modify button.
4. The system modifies the dish and redirects to the “modify dish” page.
5. TUCEW: the staff sees the modified menu.

Post-Conditions (Success Guarantee): Contents of menu have been changed.

5) Use Case 5

Name: Remove dishes

Scope: Dish Ordering System (DOS)

Level: User-goal level

Actors: Staff (Primary), DOS (supporting)

Stakeholders: Staff: deletes dishes in the menu.

Preconditions: Staff has logged in to the system.

Main Success Scenario:

1. TUCBW: the staff chooses to remove dishes.
2. The system displays the menu to staff.
3. The staff clicks on a dish.
4. The system shows a confirmation message.
5. The staff clicks on the confirm button.
6. The system saves changes and show the modified menu.
7. TUCEW: the staff sees the modified menu.

Post-Conditions (Success Guarantee): Contents of menu have been changed.

6) Use Case 6

Name: Cancel orders

Scope: Dish Ordering System (DOS)

Level: User-goal Level

Actors: Staff (Primary), DOS (supporting)

Stakeholders: Staff: cancels an existing order in the system. Customer: cancels an order he does not want.

Preconditions: The staff has logged into the system. The order which the customer wants to cancel has been made.

Main success Scenario:

1. TUCBW: the staff has logged in as admin successfully and chooses to cancel orders.
2. The system displays all orders.
3. The staff selects an order to cancel.
4. The system processes the cancellation and redirects to the order cancellation page.
5. TUCEW: the staff sees the modified order list.

Post-Conditions (Success Guarantee): The system has cancelled an order.

7) Use Case 7

Name: Log-out

Scope: Dish Ordering System (DOS)

Level: User-goal level

Actors: Staff (Primary), DOS (Supporting)

Stakeholders: Staff: logs out from the system to prevent authorization abuse.

Preconditions: The staff has logged-in to the system.

Main Success Scenario:

1. TUCBW: the staff clicks the log-out button.
2. The system logs out and redirects to the homepage.
3. TUCEW: the staff sees the homepage.

Post-Conditions (Success Guarantee): The staff has logged out from the system.

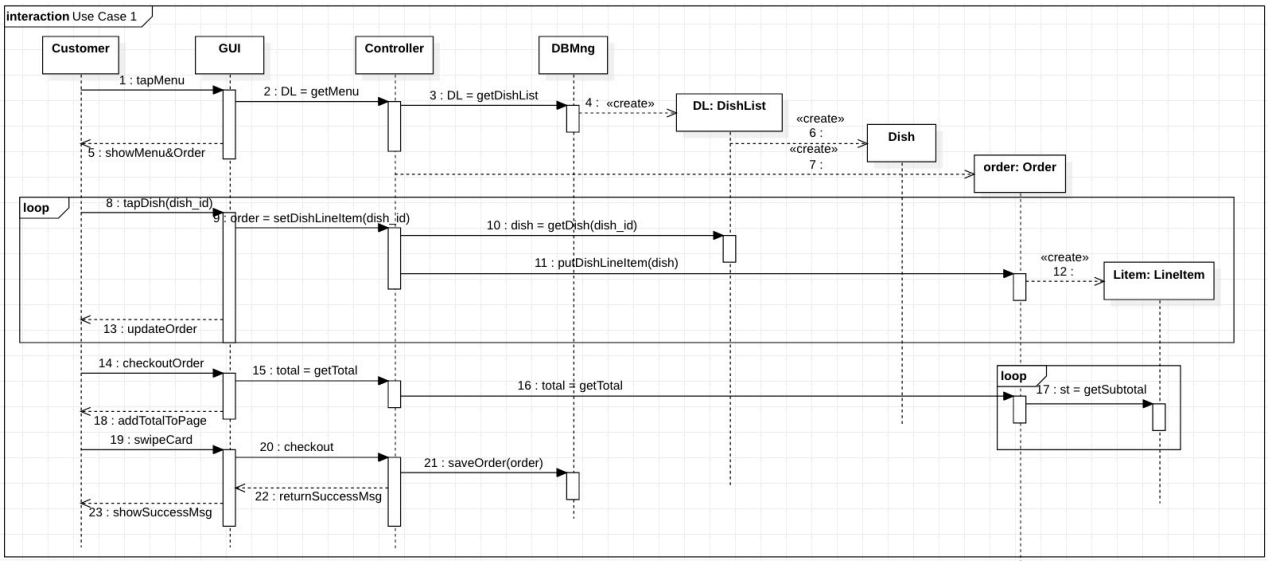
5.3 Traceability Matrix

	Priority Weight	UC1	UC2	UC3	UC4	UC5	UC6	UC7
R1	5	X						
R2	4	X						
R3	3	X						
R4	1	X						
R5	5	X						

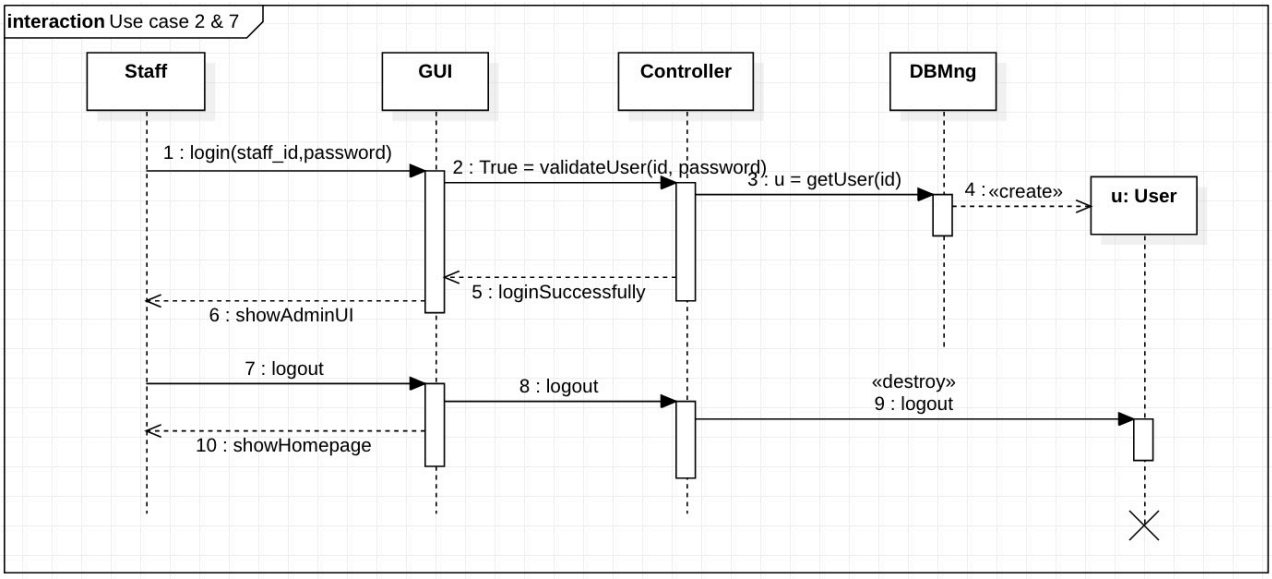
R6	3		X	X	X	X		
R7	4	X		X	X		X	
R8	5		X					X
R9	5			X	X	X		
R10	5						X	
R11	3	X						
R12	3	X						
R13	4	X	X	X	X	X	X	X
Score	50	32	12	16	16	12	13	9

6. System Sequence Diagram

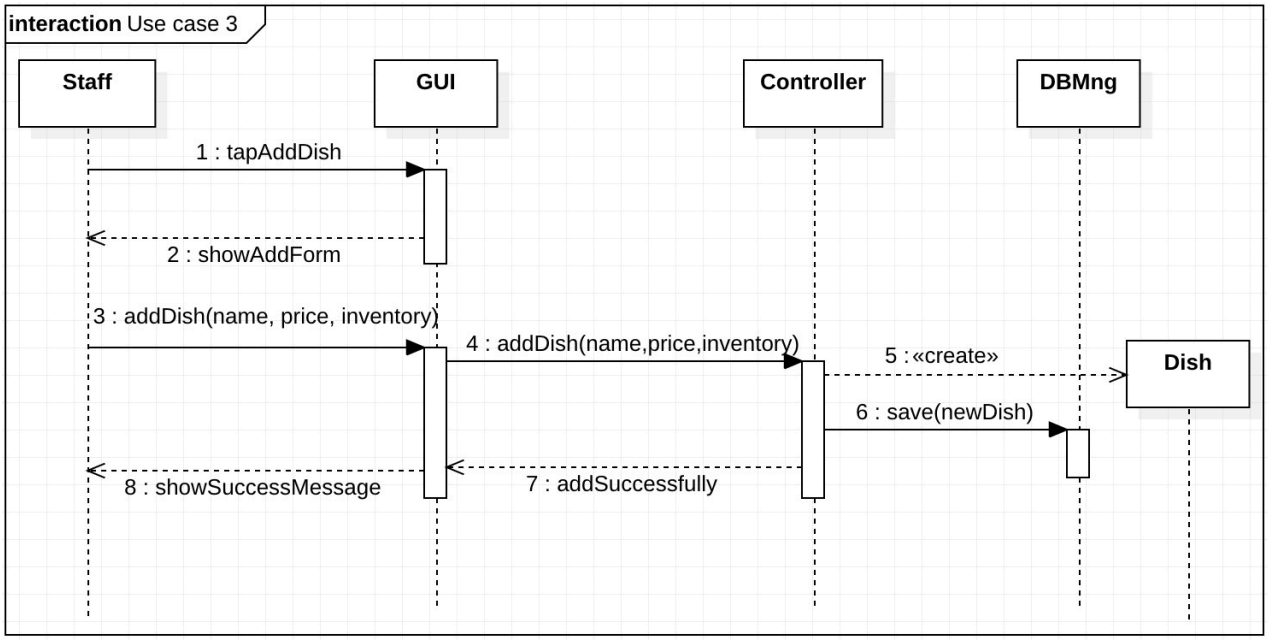
Use Case 1: Create an order



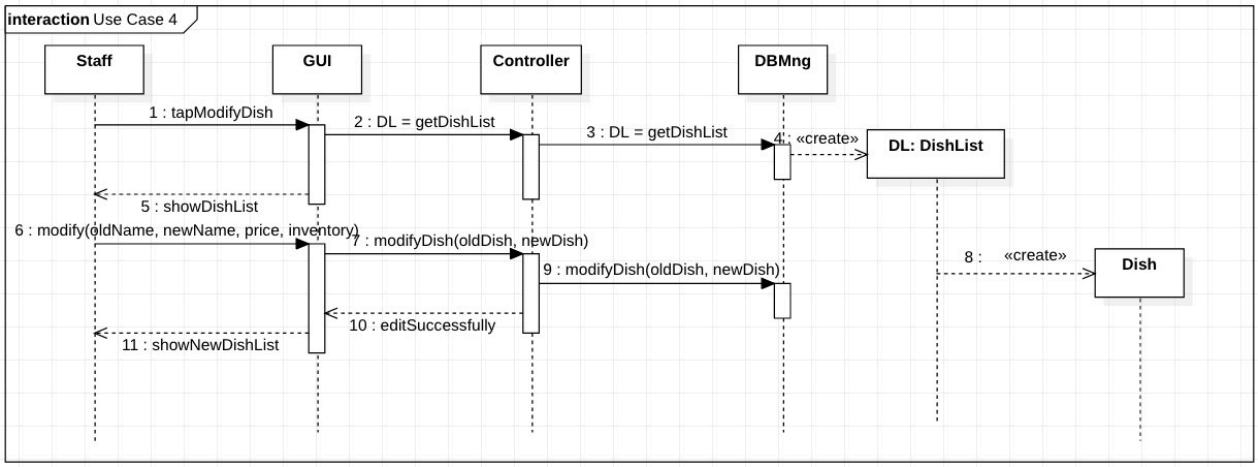
Use Case 2 & 7: Log in as admin & Log out



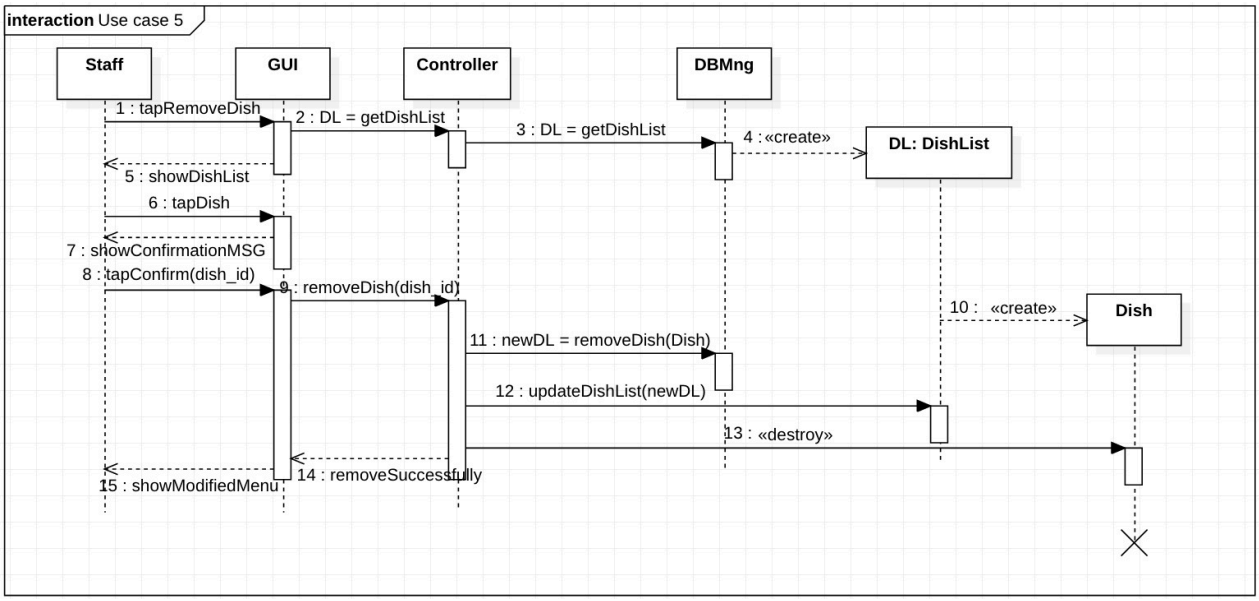
Use Case 3: Add dishes



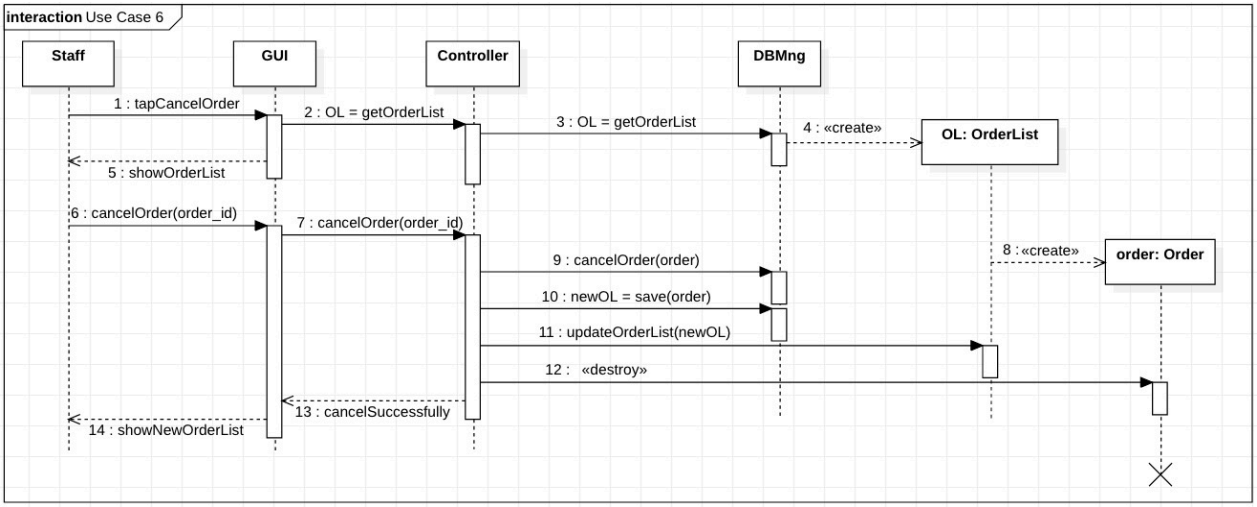
Use Case 4: Modify dishes



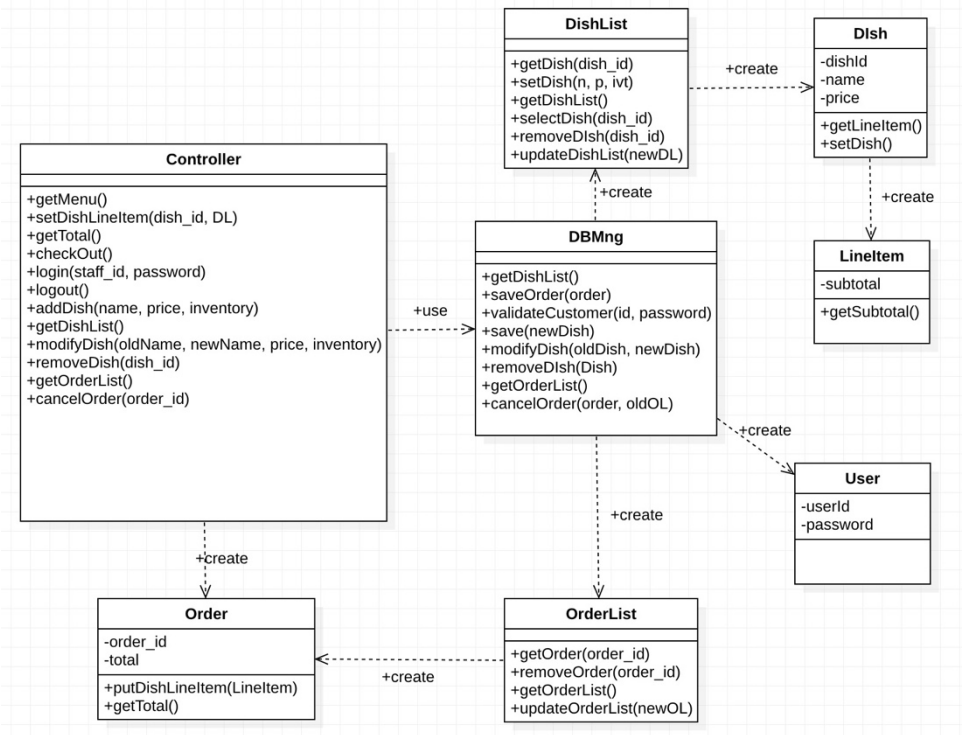
Use Case 5: Remove dishes



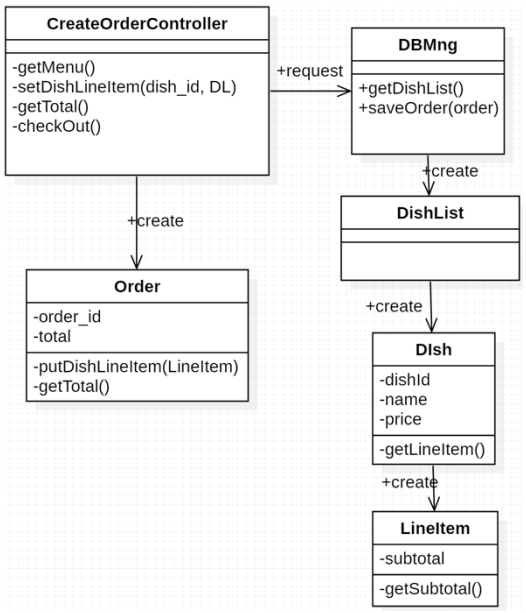
Use Case 6: Cancel orders



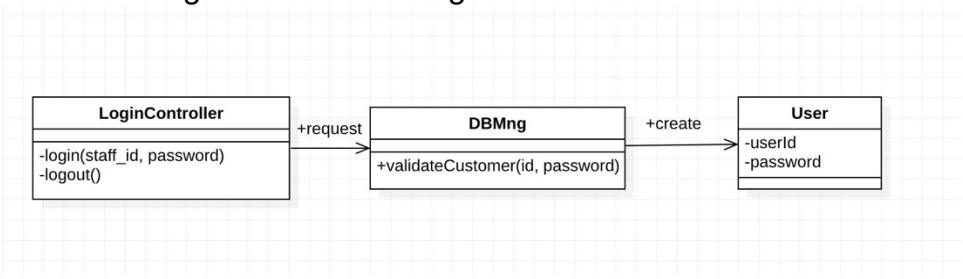
7. Class Diagram



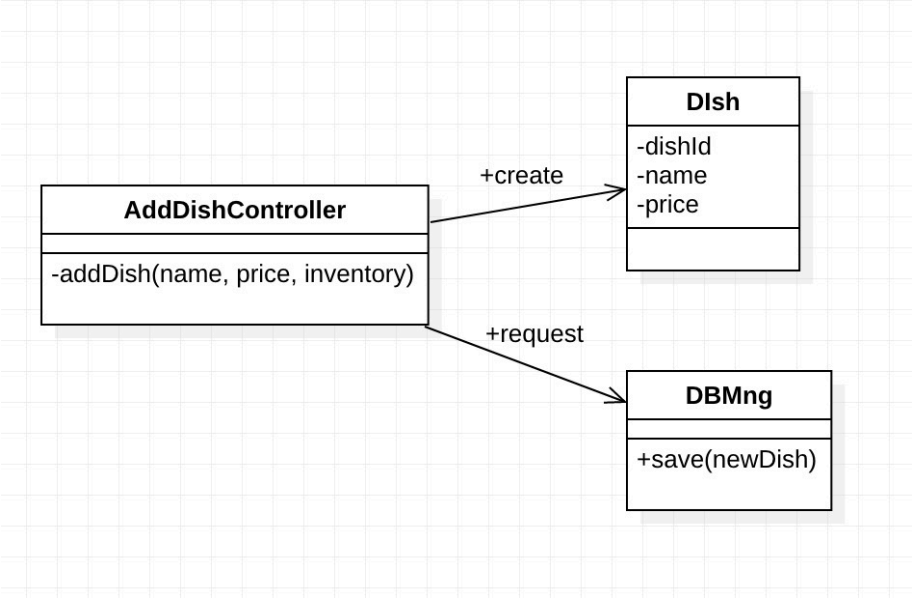
Use Case 1: Create an order



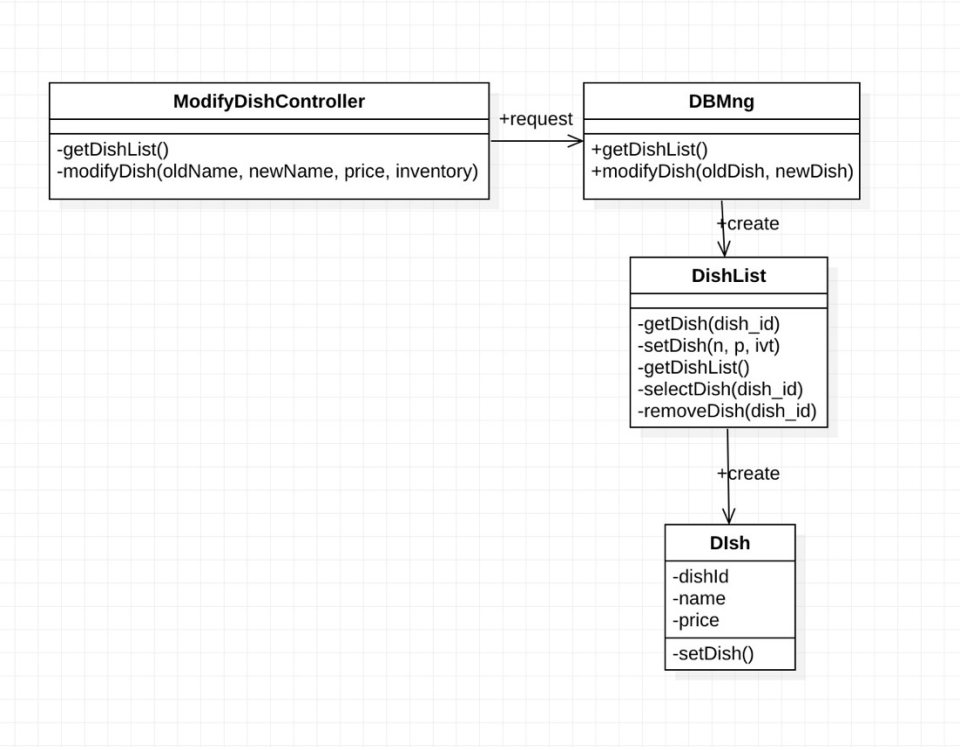
Use Case 2 & 7: Log in as admin & Log out



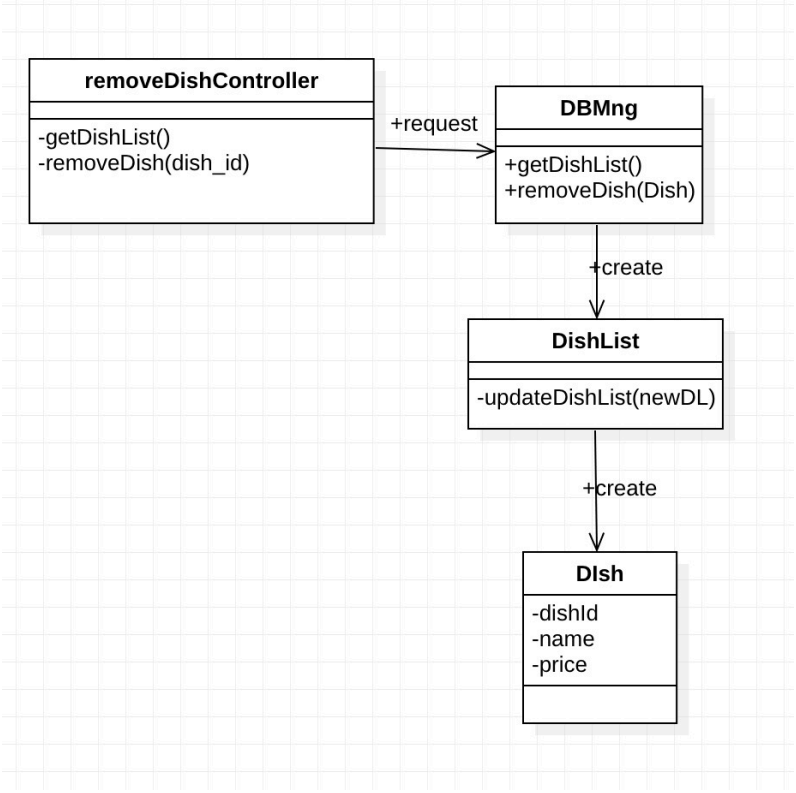
Use Case 3: Add dishes



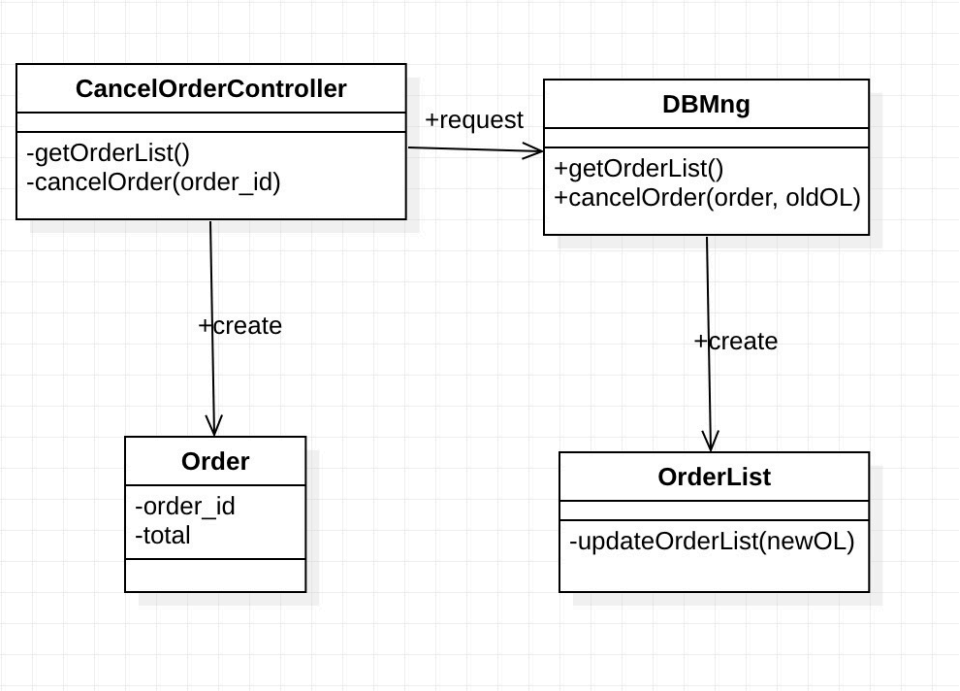
Use Case 4: Modify dishes



Use Case 5: Remove dishes



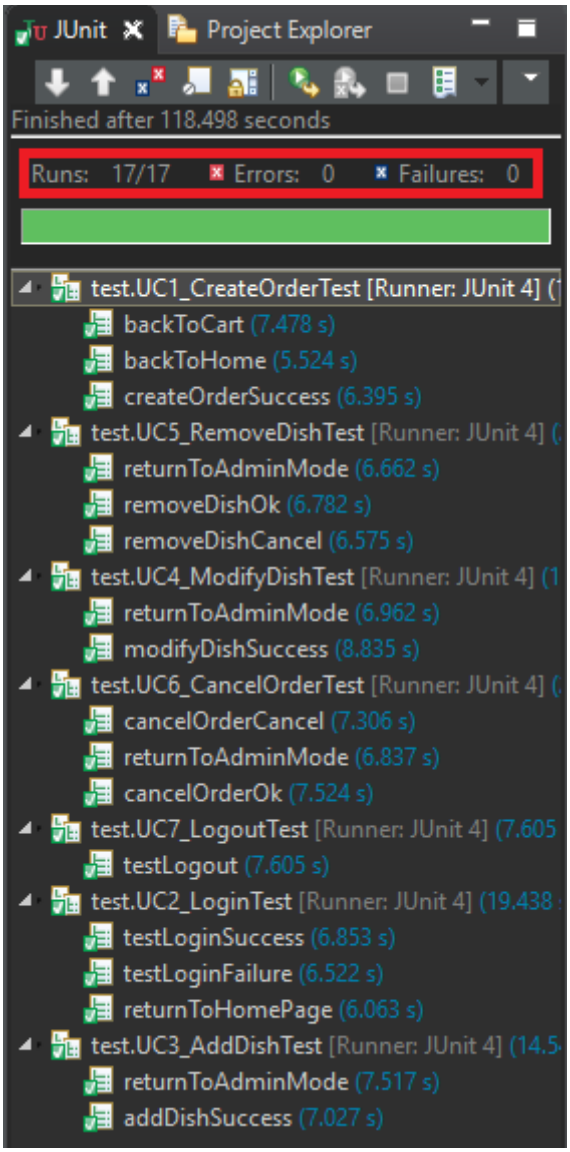
Use Case 6: Cancel orders



8. Test

Test with Junit and Selenium test. Only list non-trivial test cases. (17 Cases tested)

Test Case	Result
UC1 – createOrderSuccess()	Successfully create a new order by adding desired dishes into the order.
UC2 – testLoginSuccess()	Successfully entering admin-mode.
UC2 – testLoginFailure()	Show a login error message.
UC3 – addDishSuccess()	Successfully add a dish into the menu and can be viewed in dish modification page.
UC4 – modifyDishSuccess()	Successfully change the name, price and inventory in the menu and can be viewed in the same page.
UC5 – removeDishOk()	Successfully remove a dish from the dish list and can be checked in the same page.
UC5 – removeDishCancel()	The selected dish is not removed by clicking on cancel button.
UC6 – cancelOrderOk()	Successfully cancel and remove an order from the order list and the new order list can be viewed in the same page.
UC6 – cancelOrderCancel()	The selected order is not canceled by clicking on cancel button.
UC7 – testLogout()	Successfully logout and redirect to the home page.


JUnit
Project Explorer
Finished after 118.498 seconds
Runs: 17/17
Errors: 0
Failures: 0
test.UC1_CreateOrderTest [Runner: JUnit 4]
backToCart (7.478 s)
backToHome (5.524 s)
createOrderSuccess (6.395 s)
test.UC5_RemoveDishTest [Runner: JUnit 4]
returnToAdminMode (6.662 s)
removeDishOk (6.782 s)
removeDishCancel (6.575 s)
test.UC4_ModifyDishTest [Runner: JUnit 4]
returnToAdminMode (6.962 s)
modifyDishSuccess (8.835 s)
test.UC6_CancelOrderTest [Runner: JUnit 4]
cancelOrderCancel (7.306 s)
returnToAdminMode (6.837 s)
cancelOrderOk (7.524 s)
test.UC7_LogoutTest [Runner: JUnit 4] (7.605
testLogout (7.605 s)
test.UC2_LoginTest [Runner: JUnit 4] (19.438
testLoginSuccess (6.853 s)
testLoginFailure (6.522 s)
returnToHomePage (6.063 s)
test.UC3_AddDishTest [Runner: JUnit 4] (14.5
returnToAdminMode (7.517 s)
addDishSuccess (7.027 s)