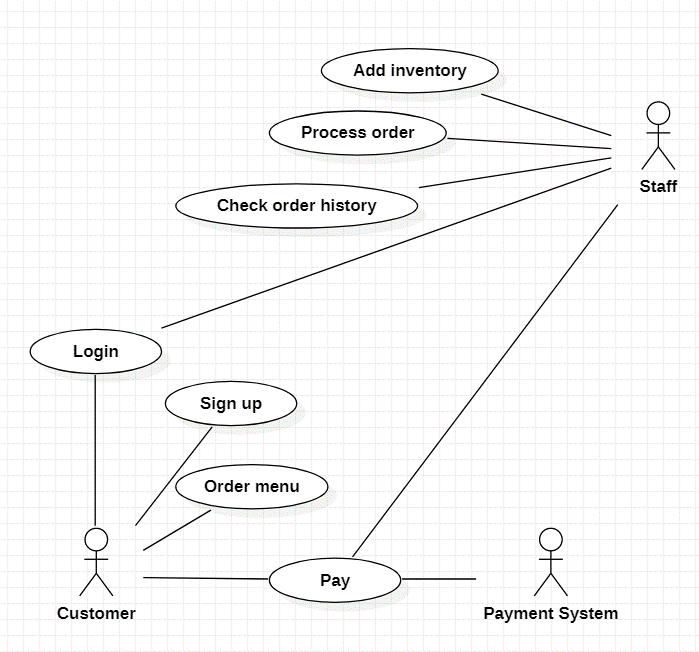
2. 유스케이스 다이어그램



액터

✓ Customer: 고객을 나타내는 액터

✓ Staff: 직원을 나타내는 액터

✓ Payment System: 결제 시스템을 나타내는 액터

유즈케이스

✓ Sign up: 고객이 회원가입을 하는 유즈케이스

✓ Login: 고객 혹은 직원이 로그인을 하는 유즈케이스

✓ Order menu: 고객이 메뉴를 주문하는 유즈케이스

✓ Pay: 고객이 주문한 메뉴에 대해 결제하는 유즈케이스

✓ Add inventory: 직원이 재고를 추가하는 유즈케이스

✓ Process order: 직원이 주문을 처리하는 유즈케이스

✓ Check order history: 직원이 주문 기록을 조회하는 유즈케이스

3. 유즈케이스 설명서와 시나리오

본 장에서는 각 유즈케이스에 대한 설명을 기술한다. 표로 구성되어있는 부분이 설명서이고 각 표 아래에 시나리오를 기술한다.

3.1. 회원가입 (Sign up)

|  |  |  |
| --- | --- | --- |
| 유즈케이스 | 회원가입 (Sign up) | |
| 유즈케이스 ID | MG-UC1 | |
| 액터 | 고객 (Customer) | |
| 요약 | 고객이 시스템에 회원으로 등록된다. | |
| 참고 유즈케이스 | 없음. | |
| 사전 조건 | 없음. | |
| 사후 조건 | (1) 고객의 회원 정보가 시스템에 등록된다. | |
| 주 흐름 | 고객 | 시스템 |
| (1) “Sign up”을 선택한다.  (3) 회원정보를 입력하고 "Submit"을 선택한다. | (2) 입력해야 하는 정보 리스트를 보여준다.  (4) 입력받은 정보를 저장한다. |
| 부 흐름 | 없음. | |
| 제약 사항 | 없음. | |

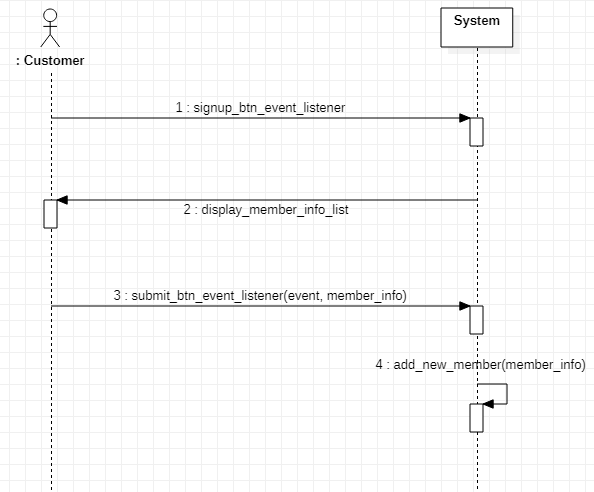
흐름

1. signup\_btn\_event\_listener()

2. display\_member\_info\_list()

3. submit\_btn\_event\_listener(event, member\_info)

4. add\_new\_member(member\_info)



3.2. 로그인 (Login)

|  |  |  |
| --- | --- | --- |
| 유즈케이스 | 로그인 (Login) | |
| 유즈케이스 ID | MG-UC2 | |
| 액터 | 고객 (Customer), 직원 (Staff) | |
| 요약 | 고객 혹은 직원이 시스템에 로그인한다. | |
| 참고 유즈케이스 | 없음. | |
| 사전 조건 | 없음. | |
| 사후 조건 | (1) 시스템이 고객 혹은 직원을 식별한다. | |
| 주 흐름 | 고객 혹은 직원 | 시스템 |
| (1) “Login"을 선택한다.  (3) id, pw를 입력하고 “Login"을 선택한다. | (2) id, pw 입력 항목을 보여준다.  (4) id, pw를 확인한 뒤, 로그인에 성공하면 id를 반환한다. 그리고 각 액터에 알맞은 메인 화면을 보여준다. |
| 부 흐름 | 4A: 입력된 id, pw가 회원 정보로 등록되어 있지 않은 경우 | |
| 고객 혹은 직원 | 시스템 |
| (2) 메시지를 확인하고 유즈케이스를 종료한다. | (1) 로그인 실패 메시지를 보여준다. |
| 제약 사항 | 없음. | |

흐름

1. login\_btn\_event\_listener()

2. display\_login()

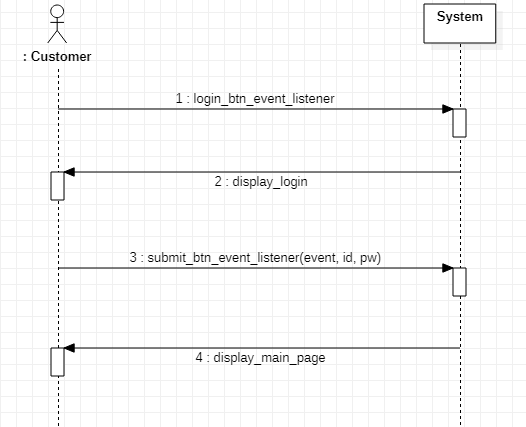
3. submit\_btn\_event\_listener(event, id, pw)

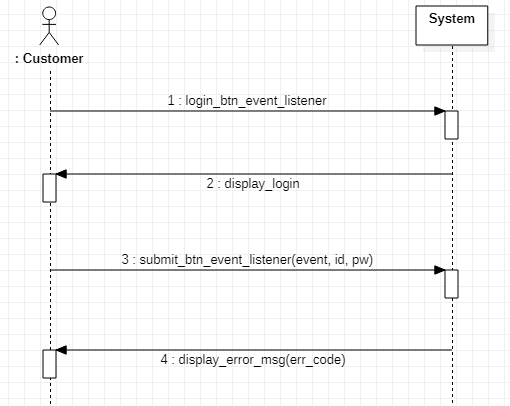
4-1. id (Customer\_view 클래스의 id 필드에 저장) -> 주문할 때 누구의 주문인지 확인 필요해서 저장함

4-2. display\_main\_page()

4A

1. display\_error\_msg(err\_code)





3.3. 주문 (Order menu)

|  |  |  |
| --- | --- | --- |
| 유즈케이스 | 주문 (Order menu) | |
| 유즈케이스 ID | MG-UC3 | |
| 액터 | 고객 (Customer) | |
| 요약 | 고객이 메뉴를 주문한다. | |
| 참고 유즈케이스 | 없음. | |
| 사전 조건 | (1) 고객이 로그인된 상태이거나 비회원 주문을 선택한 상태이다. | |
| 사후 조건 | (1) 고객이 입력한 주문 정보가 시스템에 저장된다. | |
| 주 흐름 | 고객 | 시스템 |
| (1) “메뉴”를 선택한다.  (3) 주문할 메뉴를 선택한다.  (5) “주문하기”를 선택한다.  (7) 추가 주문 정보를 입력하고 “담기”를 선택한다. | (2) 시스템에 등록된 메뉴 목록과 최근 고객이 주문한 메뉴 목록을 보여준다.  (4) 선택한 메뉴의 예시 사진과 상세 설명을 보여준다.  (6) 추가 주문 정보 입력 항목을 보여준다.  (8) 주문 정보를 장바구니에 추가한다. |
| 부 흐름 | 2A: 고객이 “비회원 주문”을 선택했거나 로그인했지만 최근 주문 내역이 없는 경우 | |
| 고객 | 시스템 |
|  | (1) 시스템에 등록된 메뉴 목록만 출력한다. |
| 4A: 고객이 선택한 메뉴가 재고 부족으로 인해 주문할 수 없는 경우 | |
| 고객 | 시스템 |
| (2) 메시지를 확인하고 유즈케이스를 종료한다. | (1) 주문 불가 메시지를 보여준다. |
| 제약 사항 | 없음. | |

흐름

1. menu\_btn\_event\_listener()

2-1. display\_menu()

2-2. display\_recent\_ordered\_menu()

3. select\_menu\_event\_listener(event)

4. display\_menu\_description(menu)

5. order\_btn\_event\_listener(event, menu)

6. display\_order\_options(menu)

7. submit\_btn\_event\_listener(event, order\_options)

8. add\_order(menu, order\_options)

4A

1. display\_error\_msg(err\_code)

\*\* order menu = display menu + display menu description + order menu

\*\* 추가 주문 정보에 들어갈 내용

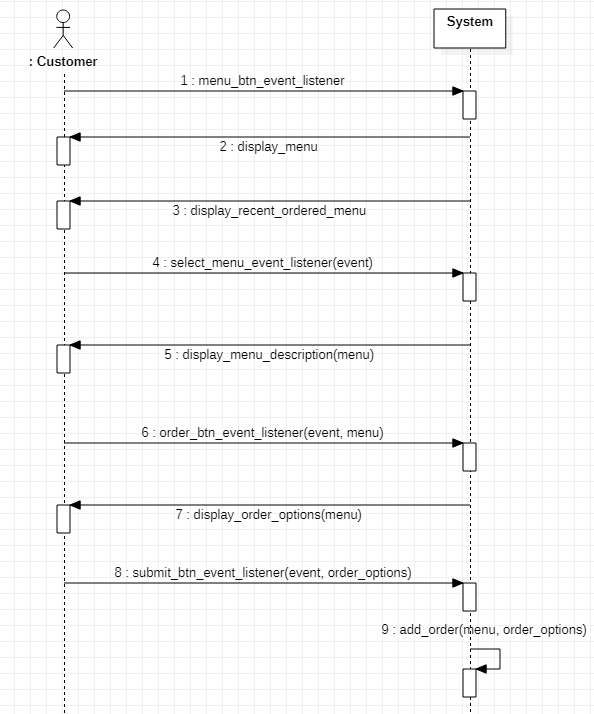
- 스타일

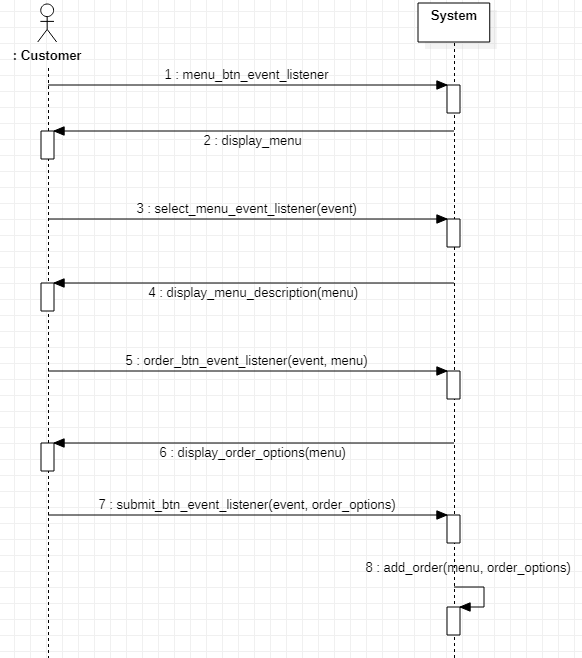
- 뺄 음식

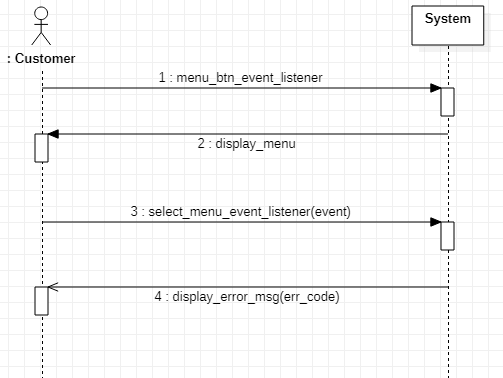
- 수량

- 배송지

- 배송 예정 시간



2A

4A

3.4. 결제 (Pay)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 유즈케이스 | 결제 (Pay) | | | |
| 유즈케이스 ID | MG-UC4 | | | |
| 액터 | 고객 (Customer), 직원 (Staff), 결제 시스템 (Payment System) | | | |
| 요약 | 고객이 주문한 메뉴에 대해 결제한다. | | | |
| 참고 유즈케이스 | 없음. | | | |
| 사전 조건 | (1) 고객의 장바구니가 최소 하나 이상의 주문을 포함해야 한다. | | | |
| 사후 조건 | (1) 시스템이 고객의 주문 정보를 저장한다. | | | |
| 주 흐름 | 고객 | 시스템 | 결제 시스템 | 직원 |
| (1) “장바구니”를 선택한다.  (3) “결제”를 선택한다.  (5) 결제 수단을 입력한다.  (7) 결제 관련 개인 정보를 입력하고 “결제”를 선택한다. | (2) 장바구니에 저장된 주문 목록을 보여준다.  (4) 결제 수단 입력 항목을 보여준다.  (6) 결제 수단에 알맞은 개인 정보 입력 항목과 총 결제 비용을 보여준다.  (8) 결제 관련 개인 정보를 결제 시스템에 전송한다.  (10) 결제 완료 메시지를 보여주고 주문 정보를 저장한다.  (11) 직원에게 새로운 주문이 있다는 메시지를 보여준다. | (9) 결제 승인 여부를 반환한다. | (12) 메시지를 확인하고 “Confirm"을 선택한다. |
| 부 흐름 | 10A: 결제가 실패한 경우 | | | |
| 고객 | | 시스템 | |
| (2) 메시지를 확인하고 유즈케이스를 종료한다. | | (1) 결제 실패 메시지를 보여준다. | |
| 제약 사항 | 없음. | | | |

흐름

1. basket\_btn\_event\_listener()

2. display\_basket()

3. pay\_btn\_event\_listener()

4. display\_pay\_options()

5. dropdown\_event\_listener()

6. display\_pay\_info()

7. submit\_btn\_event\_listener(event, pay\_info)

8. pay(pay\_info)

9. true

10. add\_order(order\_list)

11-1. display\_pay\_complete\_msg()

11-2. display\_new\_order() <- add\_order 내에서 호출하는 것

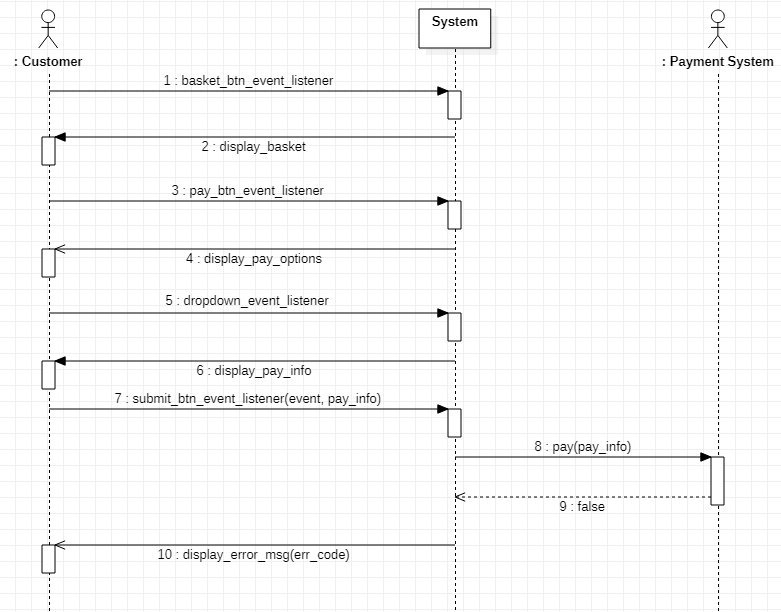
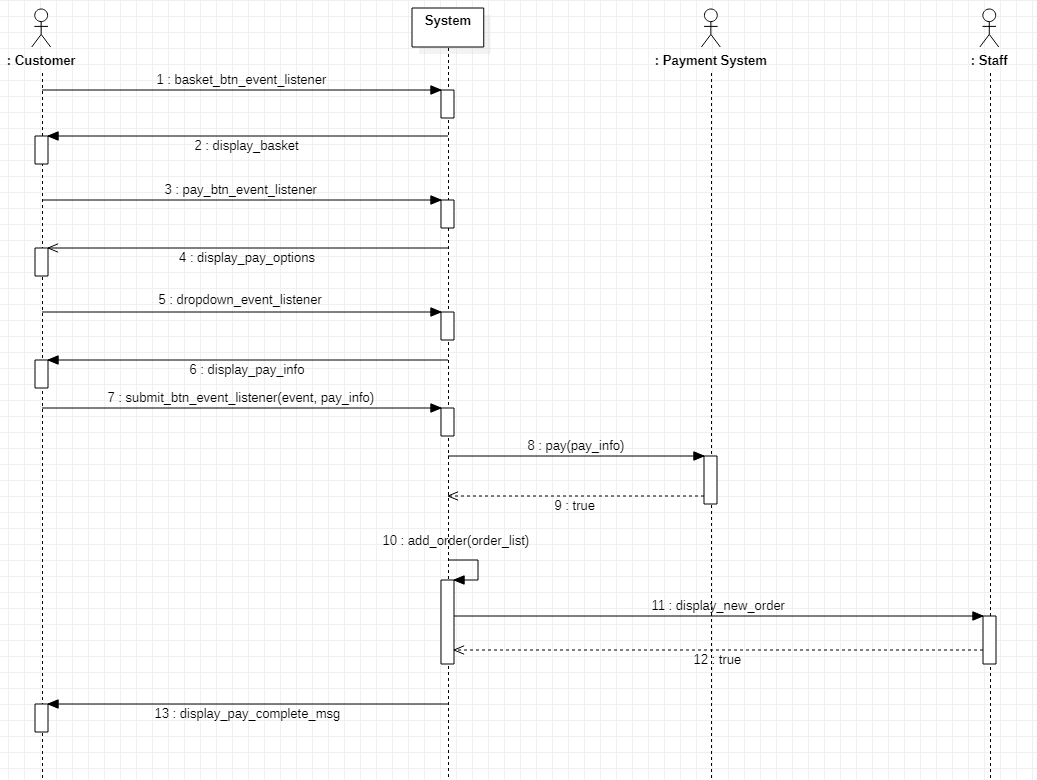
12. true

10A

0. false

1. display\_error\_msg(err\_code)

\*\* pay = display basket + pay



3.5. 재고 납품 (Add inventory)

|  |  |  |
| --- | --- | --- |
| 유즈케이스 | 재고 납품 (Add inventory) | |
| 유즈케이스 ID | MG-UC5 | |
| 액터 | 직원 (Staff) | |
| 요약 | 직원이 재고를 추가한다. | |
| 참고 유즈케이스 | 없음. | |
| 사전 조건 | (1) 직원이 로그인된 상태이다. | |
| 사후 조건 | (1) 직원이 입력한 재고를 추가하여 시스템이 반영한다. | |
| 주 흐름 | 직원 | 시스템 |
| (1) “Inventory"를 선택한다.  (3) 납품된 재고의 수량을 기존 수량 위에 입력하고 "Save"를 선택한다. | (2) 현재 재고 항목과 수량을 보여준다.  (4) 입력한 수량을 재고에 반영한다. |
| 부 흐름 | 없음. | |
| 제약 사항 | 없음. | |

\*\* add inventory = display stock + add stock

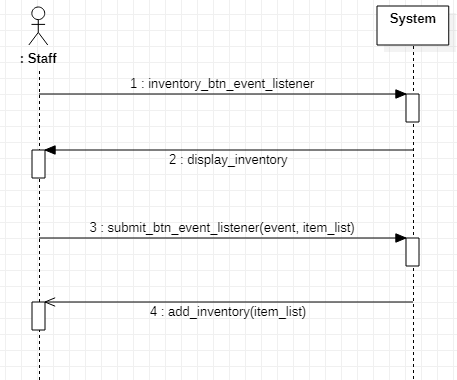
흐름

1. inventory\_btn\_event\_listener()

2. display\_inventory()

3. submit\_btn\_event\_listener(event, item\_list)

4. add\_inventory(item\_list)



~~3.6. 주문 알림 (Notice order)~~

|  |  |  |
| --- | --- | --- |
| 유즈케이스 | 주문 알림 (Notice order) | |
| 유즈케이스 ID | MG-UC6 | |
| 액터 | 직원 (Staff) | |
| 요약 | 직원이 새로운 주문을 확인한다. | |
| 참고 유즈케이스 | 결제 (MG-UC3) | |
| 사전 조건 | (1) 직원이 로그인된 상태이다.  (2) 고객이 결제를 성공하여 add\_order가 호출된다. | |
| 사후 조건 | (1) 해당 주문이 조리 중인 주문 리스트에 저장된다. | |
| 주 흐름 | 직원 | 시스템 |
| (2) "Confirm"을 선택한다. | (1) 새로운 주문에 대한 정보를 보여준다.  (3) 주문 정보를 조리 중인 주문 리스트에 저장한다. |
| 부 흐름 | 없음. | |
| 제약 사항 | 없음. | |

~~흐름~~

~~1. display\_new\_order~~

~~2. confirm\_btn\_event\_listener()~~

~~3. add\_order(order)~~

3.6. 주문 처리 (Process order)

|  |  |  |
| --- | --- | --- |
| 유즈케이스 | 주문 처리 (Process order) | |
| 유즈케이스 ID | MG-UC6 | |
| 액터 | 직원 (Staff) | |
| 요약 | 직원이 새로운 주문을 확인하고 처리한다. | |
| 참고 유즈케이스 | 없음. | |
| 사전 조건 | (1) 직원이 로그인된 상태이다.  (2) 시스템에 아직 처리되지 않은 주문이 있다. | |
| 사후 조건 | (1) 시스템이 해당 주문을 “Completed” 상태로 전환한다. | |
| 주 흐름 | 직원 | 시스템 |
| (1) “Now ordered"를 선택한다.  (3) 조리가 시작된 주문을 선택하고 "Cook"을 선택한다.  (4) 조리가 완료된 주문을 선택하고 “Deliver"를 선택한다.  (7) 배달이 완료된 주문을 선택하고 “Complete"를 선택한다. | (2) 현재 각 진행도 별로 주문 리스트를 보여준다.  (4) 선택된 주문들이 조리 중인 주문 리스트로 이동하고 정보를 갱신하여 주문 리스트를 보여준다.  (6) 선택된 주문들이 배달 중인 주문 리스트로 이동하고 정보를 갱신하여 주문 리스트를 보여준다.  (8) 선택된 주문들이 처리 완료된 주문 리스트로 이동하고 정보를 갱신하여 주문 리스트를 보여준다. |
| 부 흐름 | 없음. | |
| 제약 사항 | 없음. | |

1. now\_ordered\_btn\_event\_listener()

2. display\_order\_lists()

3. cook\_btn\_event\_listener()

4. change\_order\_status(order, COOKING)

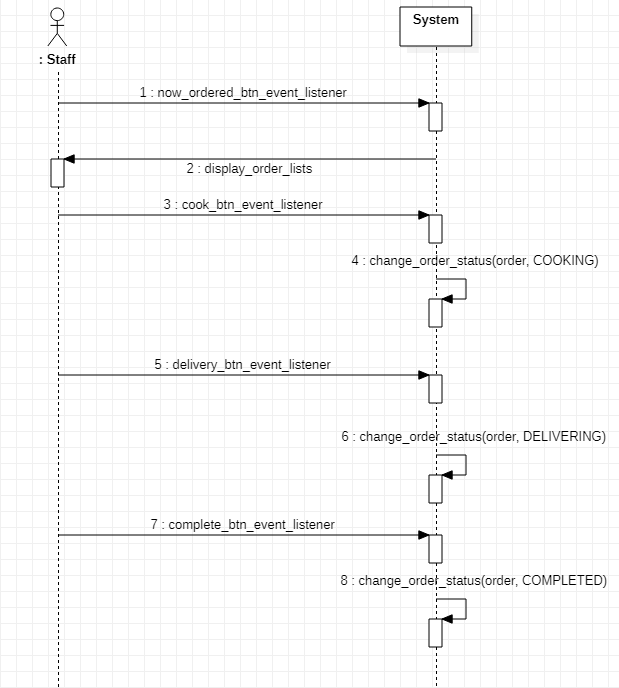
5. delivery\_btn\_event\_listener()

6. change\_order\_status(order, DELIVERING)

7. complete\_btn\_event\_listener()

8. change\_order\_status(order, COMPLETED)

※ change\_order\_status는 order, status로 두 개의 매개변수 입력을 받으며, 주문 리스트에서 체크한 것들을 뽑아서 반복문에 넣고 하나씩 처리하는 과정 (loop 체크)



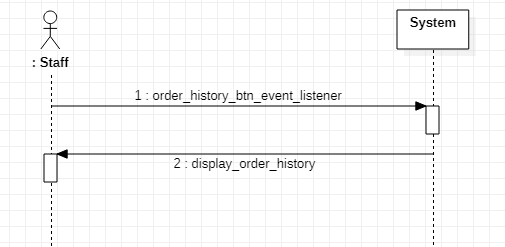
3.7. 주문 내역 확인 (Check order history)

|  |  |  |
| --- | --- | --- |
| 유즈케이스 | 주문 내역 확인 (Check order history) | |
| 유즈케이스 ID | MG-UC7 | |
| 액터 | 직원 (Staff) | |
| 요약 | 직원이 여태까지 완료된 주문을 확인한다. | |
| 참고 유즈케이스 | 없음. | |
| 사전 조건 | (1) 직원이 로그인된 상태이다. | |
| 사후 조건 | (1) 시스템이 처리된 주문의 리스트를 보여준다. | |
| 주 흐름 | 직원 | 시스템 |
| (1) “Order history"를 선택한다. | (2) 처리 완료된 주문 이력을 보여준다. |
| 부 흐름 | 없음. | |
| 제약 사항 | 없음. | |

흐름

1. order\_history\_btn\_event\_listener()

2. display\_order\_history()



3. 서브시스템 세부 설계

3.1. “Customer Client" 서브시스템 세부 설계

“Customer Client" 시스템은 고객에게 필요한 정보를 ”Server" 시스템으로부터 받아와서 보여준다. 또한, 고객으로부터 입력을 받고 "Server" 시스템으로 전송한다.

“Customer Client” 시스템의 주요 함수는 다음과 같다.

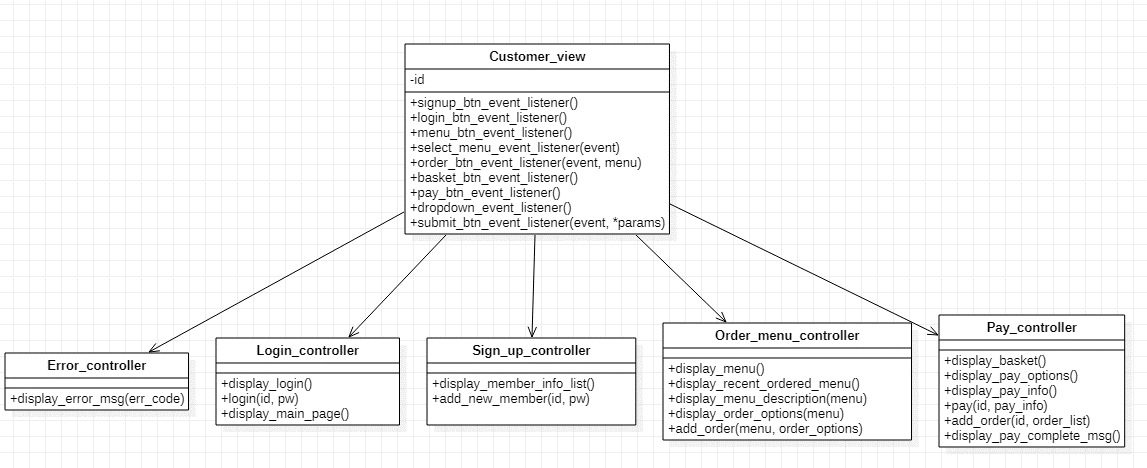
✓ 각종 display

■ "Customer Client" 시스템에서 event\_listener가 호출되었을 때 필요한 정보를 보여준다.

✓ 각종 event\_listener

■ 각종 버튼이 눌렸을 때 호출되는 함수로 "Server" 시스템에 특정 요청을 한다.

- "Customer Client" 서브시스템 메인 클래스도



Customer\_view

✓ “Customer Client" 시스템은 Customer\_view 클래스를 통해 사용자의 입력을 처리한다.

✓ Customer\_view 클래스는 입력에 따라 Controller 클래스의 함수를 호출한다.

각종 Controller

✓ “Customer Client" 시스템은 각종 Controller 클래스를 통하여 “Server" 시스템에 요청을 보낸다.

3.2. “Staff Client" 서브시스템 세부 설계

"Staff Client" 시스템은 주문 내역과 재고 내역을 “Server" 시스템으로부터 받아와서 보여준다. 그리고 납품으로 늘어난 재고의 양과 완료된 주문 목록을 입력으로 받고 ”Server" 시스템으로 전송한다.

“Staff Client" 시스템의 주요 함수는 다음과 같다.

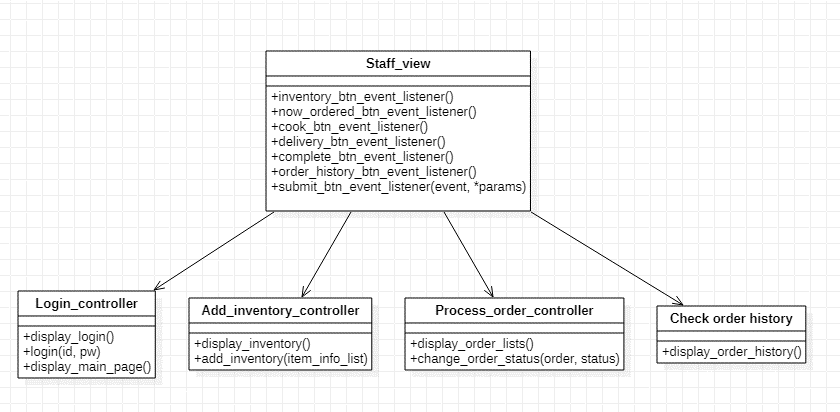
✓ 각종 display

■ "Staff Client" 시스템에서 event\_listener가 호출되었을 때 필요한 정보를 보여준다.

✓ 각종 event\_listener

■ 각종 버튼이 눌렸을 때 호출되는 함수로 "Server" 시스템에 특정 요청을 한다.

- "Customer Client" 서브시스템 메인 클래스도



Staff\_view

✓ “Staff Client" 시스템은 Staff\_view 클래스를 통해 사용자의 입력을 처리한다.

✓ Staff\_view 클래스는 입력에 따라 Controller 클래스의 함수를 호출한다.

각종 Controller

✓ “Staff Client" 시스템은 각종 Controller 클래스를 통하여 “Server" 시스템에 요청을 보낸다.

3.3. “Server" 서브시스템 세부 설계

“Server" 시스템은 고객에게 메뉴 관련 정보를 제공하고 주문 정보를 저장한다. 또한, "Customer Client”에게 받은 요청을 처리하는 역할을 한다.

“Server" 시스템의 주요 함수는 다음과 같다.

✓ signup(id, pw)

■ “Customer Client"에서 새로운 회원 정보를 입력받아 저장한다.

✓ login(id, pw)

■ "Customer Client" 혹은 “Staff Client"에서 로그인 정보를 입력받아 사용자를 식별한다.

✓ get\_recent\_ordered\_menu\_list()

■ 로그인한 고객에 대해 최근 주문 내역을 반환한다.

✓ add\_order(order)

■ "Customer Client" 시스템으로부터 전송받은 정보로 주문 정보를 생성하고 장바구니에 추가한다.

✓ pay(pay\_info)

■ 결제 시스템에게 결제 정보를 넘기며 결제를 요청한다.

✓ get\_item\_list()

■ 현재 남은 재고의 수량을 반환한다.

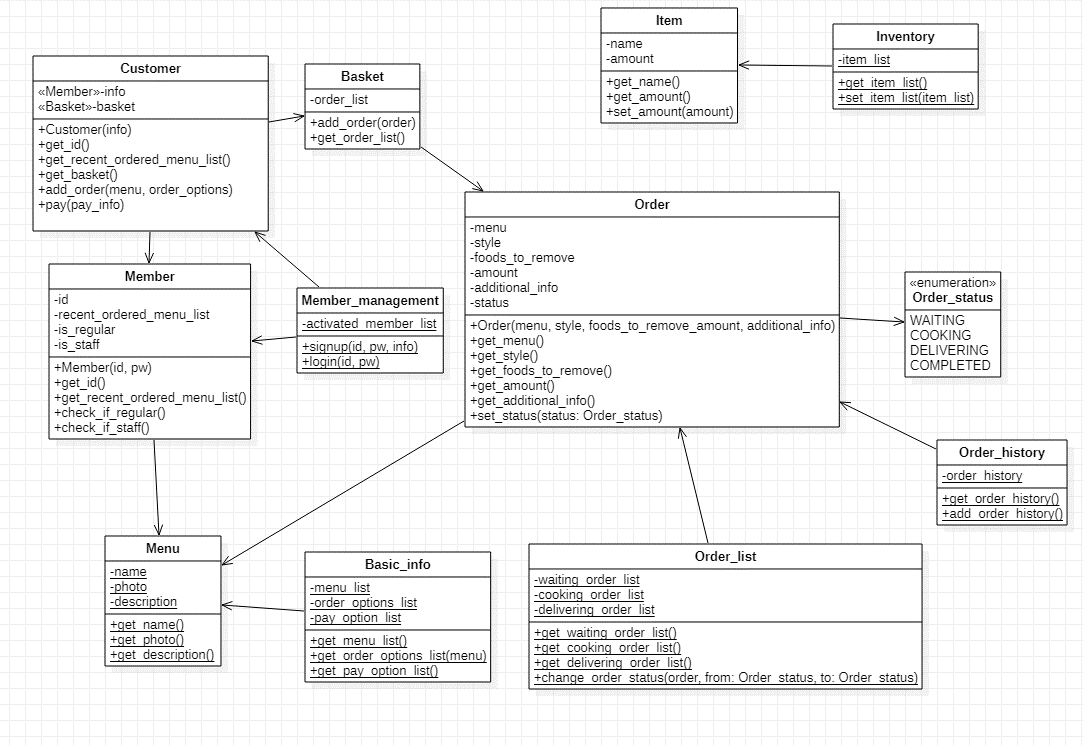
✓ get\_order\_history()

■ 최근 서른 개의 주문 내역을 반환한다.

✓ change\_order\_status(order, from, to)

■ 지정한 order의 상태를 from에서 to로 바꾼다.

- "Server" 시스템 메인 클래스도



Basic\_info

✓ 미스터 대박 서비스의 기본적인 정보를 담고 있는 클래스이다.

✓ "Server" 시스템은 Basic\_info를 통해 "Customer Client" 시스템에게 메뉴 정보, 메뉴에 따른 주문 옵션 정보, 결제 정보를 넘겨준다.

Menu

✓ 메뉴 정보를 저장하는 type을 나타내는 클래스이다.

✓ 예시 사진과 메뉴 설명에 대한 정보를 담고 있다.

Member\_management

✓ 회원가입과 로그인을 관리하는 클래스이다.

✓ 회원가입과 로그인을 수행할 수 있으며, 현재 시스템에 로그인된 고객의 리스트를 속성으로 갖는다.

Member

✓ 로그인한 유저를 식별하는 클래스로 id, 최근 주문 내역, 단골 고객 여부, 직원 여부를 속성으로 갖는다.

✓ 로그인 시에 객체가 생성되며 이 때 생성된 객체를 토대로 고객과 직원을 식별하고, 서로 다른 고객을 식별한다.

Customer

✓ 고객을 나타내는 클래스이다.

✓ 고객을 식별하는 Member 객체, 장바구니를 속성으로 갖는다.

✓ pay 함수를 통해 장바구니에 담긴 주문에 대해 최종 결제를 할 수 있다.

Basket

✓ 장바구니 type 클래스이다.

✓ 장바구니는 주문 정보의 집합이며 add\_order를 통해 장바구니에 주문 정보를 추가할 수 있다.

Order

✓ 주문 정보를 저장하는 type을 나타내는 클래스이다.

✓ 메뉴, 스타일, 뺄 음식, 수량 정보, 추가 주문 정보를 저장하고 있으며 생성자로 주문 정보를 설정할 수 있다.

Order\_list

✓ 아직 완료되지 않은 주문을 관리하는 클래스이다.

✓ 상태별로 클래스를 리스트로 저장하고 있으며, change\_order\_status 함수로 각 주문의 상태를 전환 가능하다.

Order\_history

✓ 주문 내역을 관리하는 클래스이다.

✓ 주문 내역을 저장하고 있으며, 주문이 완료될 시 order\_history에 저장하는 역할을 한다.

Order\_status

✓ 주문의 상태를 나타내는 enumeration으로 네 가지의 상태를 나타내는 데 사용된다.

Inventory

✓ 재고를 나타내는 클래스이다.

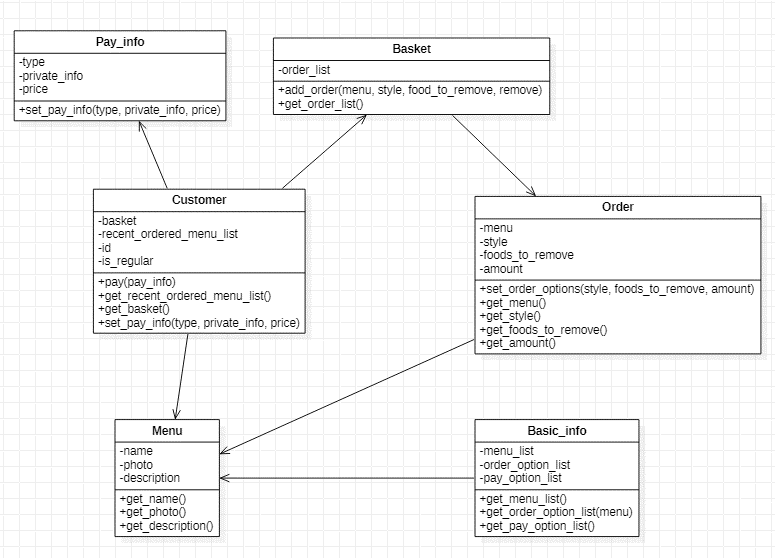
✓ 재고 목록을 속성으로 갖고 있으며 set\_item\_list 함수를 통해 재고 수량을 수정할 수 있다.

Item

✓ 재고에 속하는 재료의 type을 나타내는 클래스이다.

✓ 이름과 현재 보유 수량을 속성으로 갖고 있다.

2. “Server" 시스템



Basic\_info

✓ 미스터 대박 서비스의 기본적인 정보를 담고 있는 클래스이다.

✓ "Server" 시스템은 Basic\_info를 통해 "Customer Client" 시스템에게 메뉴 정보, 메뉴에 따른 주문 옵션 정보, 결제 정보를 넘겨준다.

Menu

✓ 메뉴 정보를 저장하는 type을 나타내는 클래스이다.

✓ 예시 사진과 메뉴 설명에 대한 정보를 담고 있다.

Customer

✓ 고객을 나타내는 클래스이다.

✓ 고객을 식별하는 id, 단골 고객 여부, 장바구니, 최근 주문 내역을 저장하고 있다.

✓ Pay 함수를 통해 장바구니에 담긴 주문에 대해 최종 결제를 할 수 있다.

Basket

✓ 장바구니 type 클래스이다.

✓ 장바구니는 주문 정보의 집합이며 add\_order를 통해 장바구니에 주문 정보를 추가할 수 있다.

Order

✓ 주문 정보를 저장하는 type을 나타내는 클래스이다.

✓ 메뉴, 스타일, 뺄 음식, 수량 정보를 저장하고 있으며 set\_order\_options로 주문 정보를 설정할 수 있다.

Pay\_info

✓ 결제 정보를 저장하는 type을 나타내는 클래스이다.

✓ 결제 수단, 결제 관련 개인 정보, 최종 결제 가격 정보를 저장하고 있으며 set\_pay\_info로 결제 정보를 설정할 수 있다.