Online Pizza Ordering System

**<SDD – The 2nd Checkpoint>**

소프트웨어공학 02분반

20171105 이민욱

20173875 정용준

20170223 신원준

20172609 여일구

20174438 정종민

20173156 김준기

목차

[Online Pizza Ordering System 1](#_Toc71665406)

[**<SDD – The 2nd Checkpoint>** 1](#_Toc71665407)

[**1.** **SubgroupA(PizzaOrder)** 2](#_Toc71665408)

[1.1 **A-UC-1 blah blah** 2](#_Toc71665409)

[**2.** **SubgroupB(Management)** 2](#_Toc71665410)

[2.1 **B-UC-1: AuthenticateUser** 2](#_Toc71665411)

[2.1.1 **최종 결과 및 선정 이유** 2](#_Toc71665412)

[2.2 **B-UC-3: AddPizza** 3](#_Toc71665413)

[2.3 **B-UC-9: DeleteUser** 3](#_Toc71665414)

[2.4 **DisplayByPeriod** 3](#_Toc71665415)

[**3.** **SubgroupC(OrderStatus)** 3](#_Toc71665416)

[3.1 **C-UC-1 blah blah** 3](#_Toc71665417)

# **SubgroupA(PizzaOrder)**

* 1. **A-UC-1 blah blah**

# **SubgroupB(Management)**

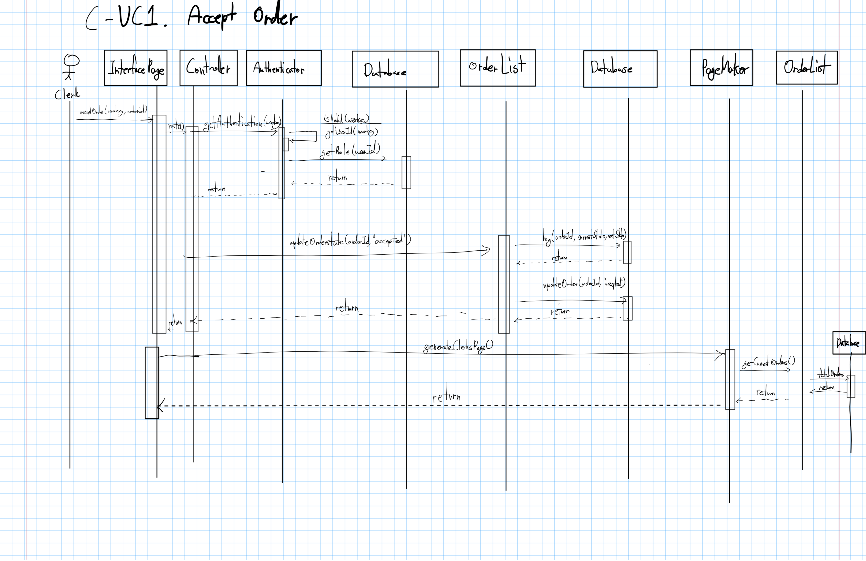
* 1. **B-UC-1: AuthenticateUser**

1. Basic
2. Variation1
3. Variation 2
4. Variation 3
   * 1. **최종 결과 및 선정 이유**
5. Object Sequence Diagram
6. Class Diagram
7. Traceability Matrix
   1. **B-UC-3: AddPizza**
   2. **B-UC-9: DeleteUser**
   3. **DisplayByPeriod**

# **SubgroupC(OrderStatus)**

* 1. **C-UC-1 Accept Order**

1. Variation1



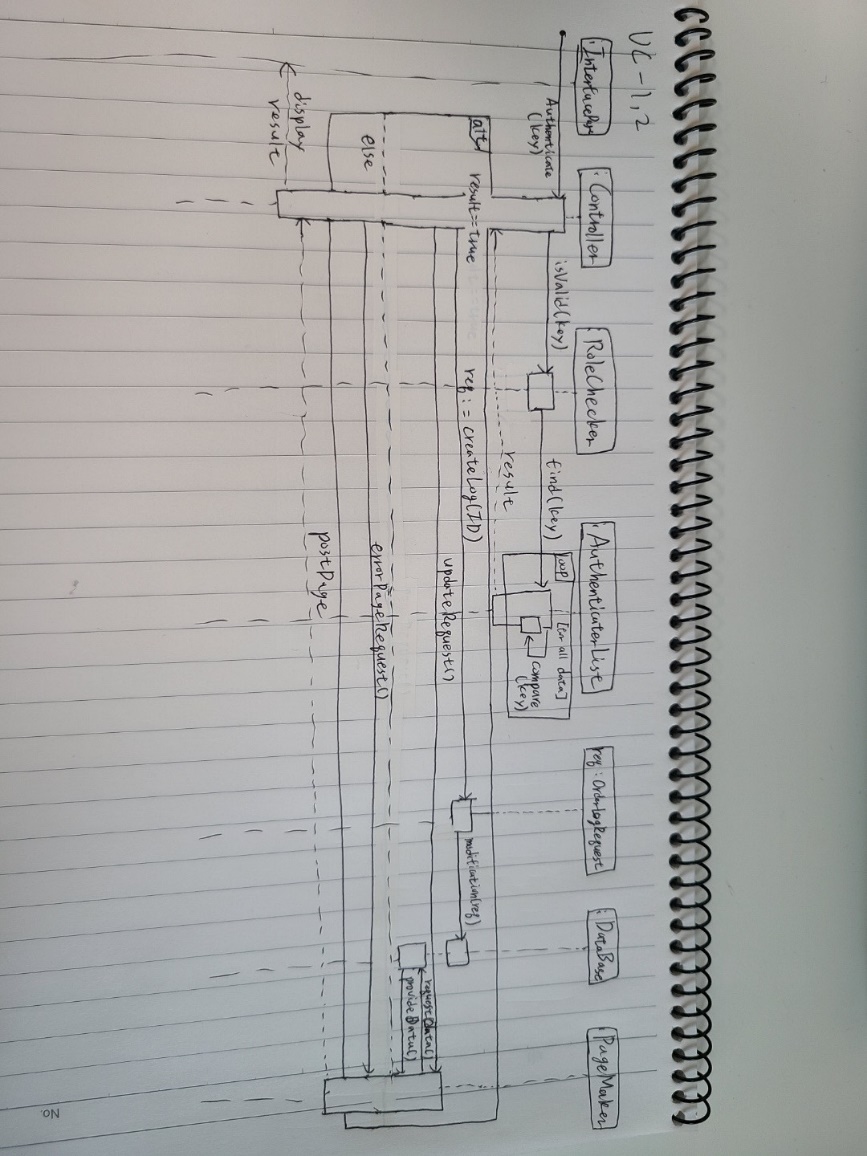
어떤 user로부터 주문을 accept하겠다는 요청이 들어오면, 우선 해당 user의 userkey를 받아서 Authenticator에서 이 user가 Clerk인지를 체크한다. Authenticator 내부에서 userkey를 통해 userID를 찾고, Database에서 해당 userID와 맵핑되는 role을 받아와서 이 role이 Clerk인지를 확인한다.

Clerk이 맞다면, orderID와 이 주문이 accept 되었다는 것을 OrderList에 전해주고, OrderLIst는 log와 updateOrder를 통해 log를 데이터베이스에 남긴 후에 실질적인 주문현황을 최신화해준다.

이러한 작업이 모두 끝나면, InterfacePage에서 PageMaker에 새로운 페이지를 요청하고, OrderList, Database를 거쳐 가장 최근에 업데이트 된 주문현황을 받아와 페이지를 재구성한다.

하지만 이는 InterfacePage에 너무 많은 정보가 노출되고, InterfacePage는 Thing에 해당하는 객체이기 때문에 일을 직접적으로 하는 것이 아닌 전달요소로만 쓰여야한다. 따라서 이러한 페이지 요청이 Controller가 요청하는 것으로 되어야 한다.

1. Variation2

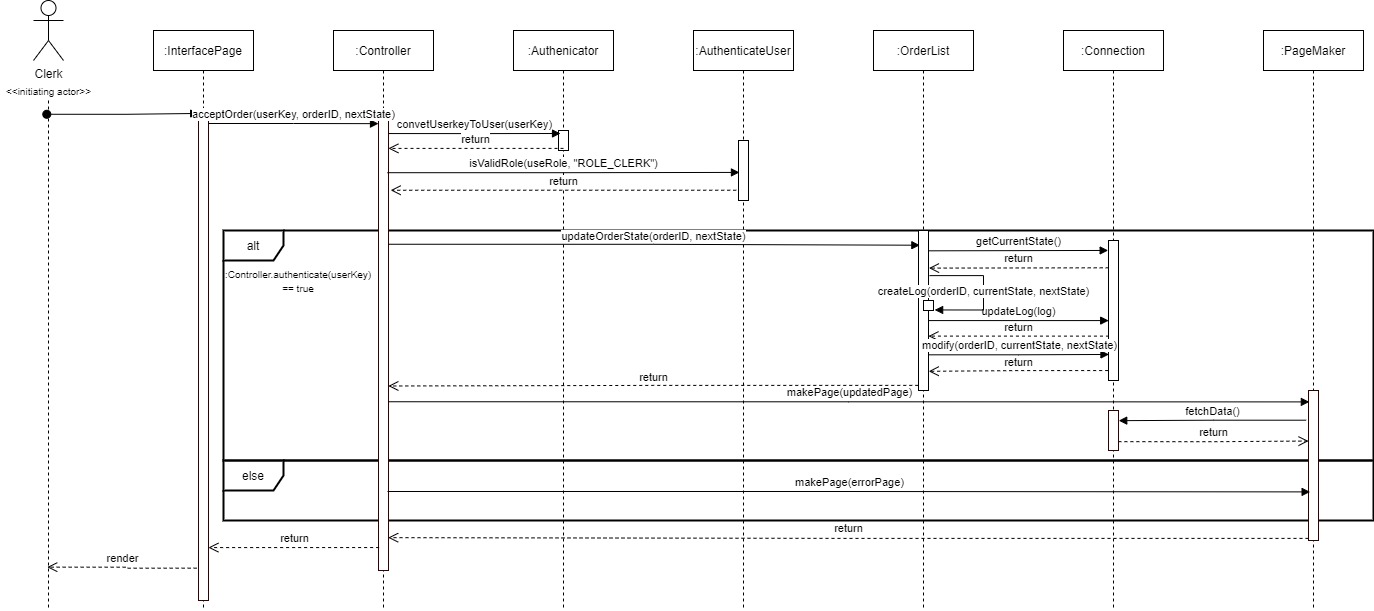


accept 요청이 들어오면, 이때 user의 key를 같이 받아와 RoleChecker에게 전달해준다. RoleChecker는 AuthenticaterList에서 전달받은 key에 맵핑된 role이 accept 작업을 수행할 수 있는지 compare한 후, 이를 True/False로 전달해준다.

True일 경우, Log를 만들어 이를 Database에 업데이트 해준다. 이후 PageMaker에 페이지를 업데이트 해줄 것을 요청하고, PageMaker는 필요한 정보를 Database에서 받아와 업데이트를 해준다. False일 경우, PageMaker는 Database에서 아무것도 받지 않고 에러창을 생성하여 띄워준다.

응답에 따라 분기를 하는 것이 잘 표현되어 있지만, return이 없는 메서드가 존재하는 등의 문제가 있다. 또한 메서드명이 불분명하여 이를 재명명하는 것이 필요하다.

* + 1. **최종 결과 및 선정 이유**



Variation2를 base로 채택하였다.

Variation2에서 isValid()같은 역할이 모호한 함수를 수정하고 return이 없는 함수들에 return 추가하였다.

사용자가 주문을 수락할 권한이 있는가를 판단할 때, Authenticator가 userKey에 대응되는 userRole을 생성하여 userRole이 ROLE\_CLERK과 일치하는지 비교하도록 변경하였다.

currentState를 Clerk이 상태를 변화시킬 때 전달할 필요가 없기 때문에, 요청에서 이러한 내용이 빠지고, currentState를 Connection로부터 가져오는 부분이 추가되었다. 또한 OrderList가 Log를 남기는 것과 상태를 업데이트 하는 것이 구체화되었다.

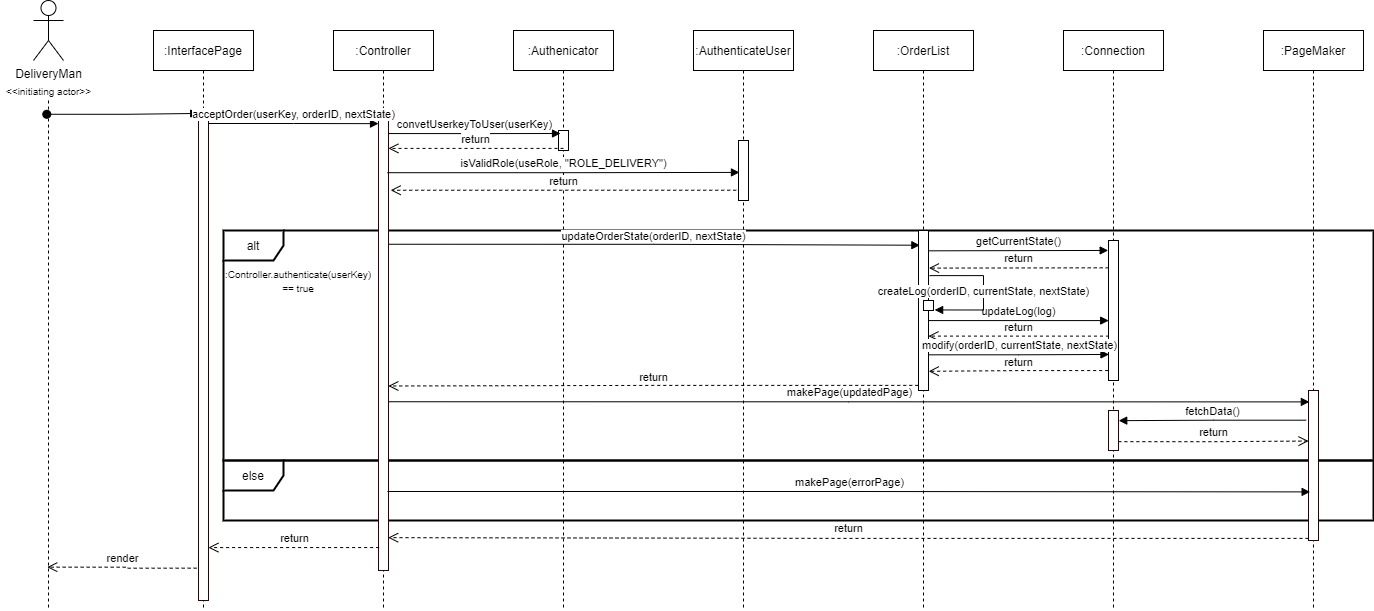
InterfacePage는 정보를 제공만 할 뿐 어떠한 행동을 하는 Responsibility는 Controller로 옮겼다.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Domain Conecpts | Software class | | | | | | |
| InterfacePage | Controller | Authenticator | AuthenticateUser | OrderList | Connection | PageMaker |
| InterfacePage | O |  |  |  |  |  |  |
| Controller |  | O |  |  |  |  |  |
| Authenticator |  |  | O | O |  |  |  |
| OrderList |  |  |  |  | O | O | O |
| PageMaker |  |  |  |  |  |  | O |

* 1. **C-UC-2 Complete Cook**

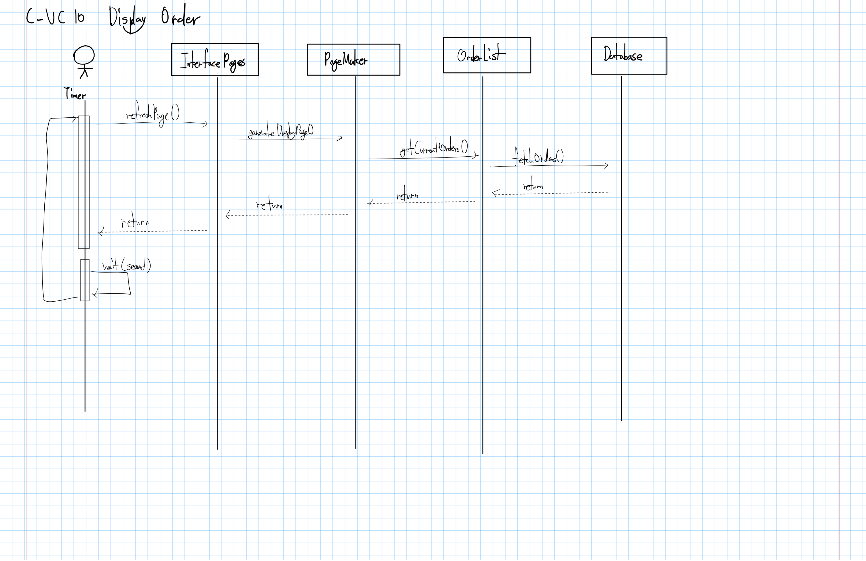
Initiating actor가 Clerk에서 DeliveryMan으로 바뀌는 것을 제외하면, C-UC-1과 전부 동일하다.

* + 1. **최종 결과 및 선정 이유**



* 1. **C-UC-10 Display Order**

1. Variation1



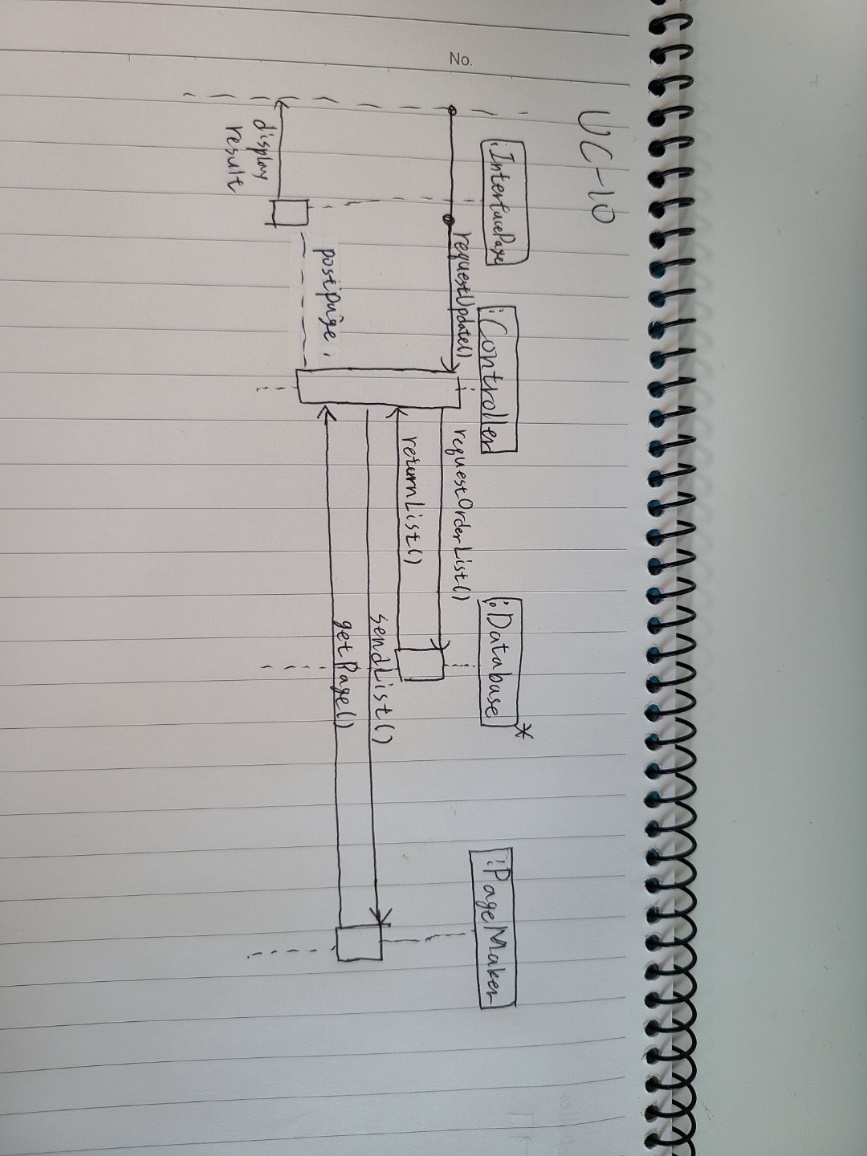
매장에 설치되어 있는 Screen에 대한 usecase이므로, 주기적인 업데이트가 필요하다고 판단되어 Initiating actor를 Timer를 만들어 설정하였다.

Timer가 refreshPage()를 호출하면, PageMaker에서 OrderList로, OrderList에서 Database로 이동하며 화면을 만드는데 필요한 정보들을 요청하고, return된 정보들을 통해 PageMaker에서 새로운 페이지를 만들어 업데이트해준다.

이 작업이 끝나면, 주기적으로 업데이트를 해주기 위해, Timer에서 약간의 시간을 wait()한 이후 위의 과정을 반복한다.

이 그림에서 Timer는 어떻게 시작되며, 언제 종료되는지가 불명확하다. 또한 이렇게 생성된 InterfacePage를 최종적으로 누구에게 제공하는지가 없다.

1. Variation2

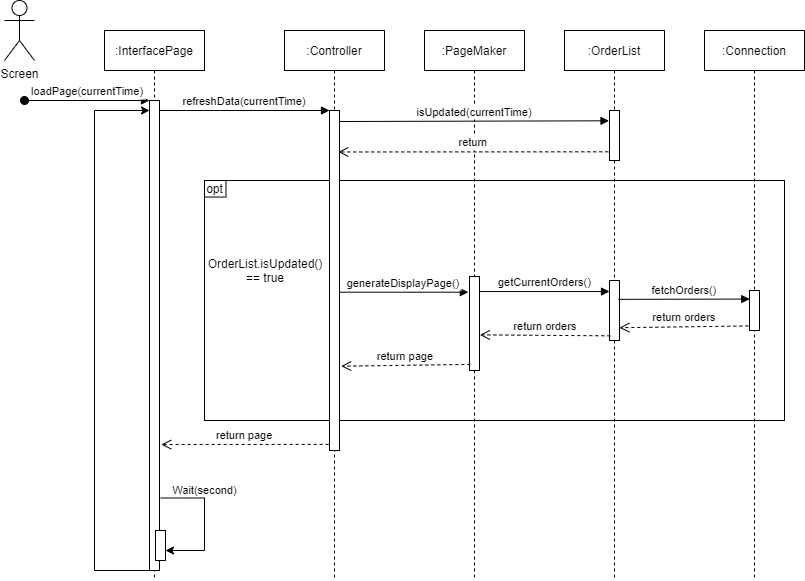


Screen이 Controller에게 페이지를 업데이트할 것을 요청하고, Controller는 페이지를 구성하는데 필요한 정보를 Database에서 가져와 이를 PageMaker에 전달한다.

PageMaker는 이를 바탕으로 새로운 페이지를 생성하여 InterfacePage에 전달해준다.

이를 Screen에 띄우는 것으로 추정되는데 이것이 명확하게 표현되지 않고 있고, 주기적으로 update가 되어야하는 페이지임에도 불구하고 이를 명시한 흐름을 찾아보기 어렵다.

* + 1. **최종 결과 및 선정 이유**



Variation1을 base로 채택하였다.

orderList가 업데이트 될 때 마다 혹은 주기적인 시간마다 Screen이 업데이트 되어야 하는데, Variation2에서는 이러한 사항이 배제되어 있다.

또한 Variation2는 cohesion을 과도하게 의식하여 전체적인 flow가 비효율적이고 Controller가 unrelated한 responsibility를 가지고 있다고 판단하였다.

Variation1의 전체적인 flow를 OrderList.isUpdated() == true인 부분에 사용하였고, 기존에 있던 usecase 설계를 반영하여 actor를 Screen으로 변경하고 Controller object를 추가하였다.

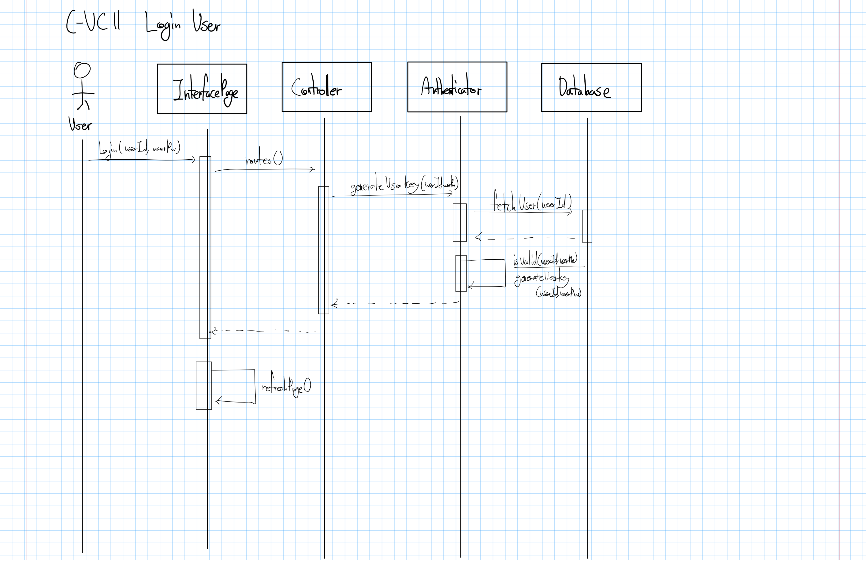
Controller는 refreshData()를 통해 currentTime을 받아오고, OrderList는 isUpdated()를 통해 currentTime 이후에 orderList에 변동이 생겨 페이지 업데이트가 필요한지 여부를 판단한다.

Controller가 returnPage()를 하면 InterfacePage는 wait()을 한 후, 다시 loadPage()를 호출하도록 하여 주기적으로 화면 업데이트를 요청하도록 하였다.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Domain Conecpts | Software class | | | | |
| InterfacePage | Controller | OrderList | Connection | PageMaker |
| InterfacePage | O |  |  |  |  |
| Controller |  | O |  |  |  |
| OrderList |  |  | O | O |  |
| PageMaker |  |  |  |  | O |

* 1. **C-UC-11 Login User**

1. Variation1

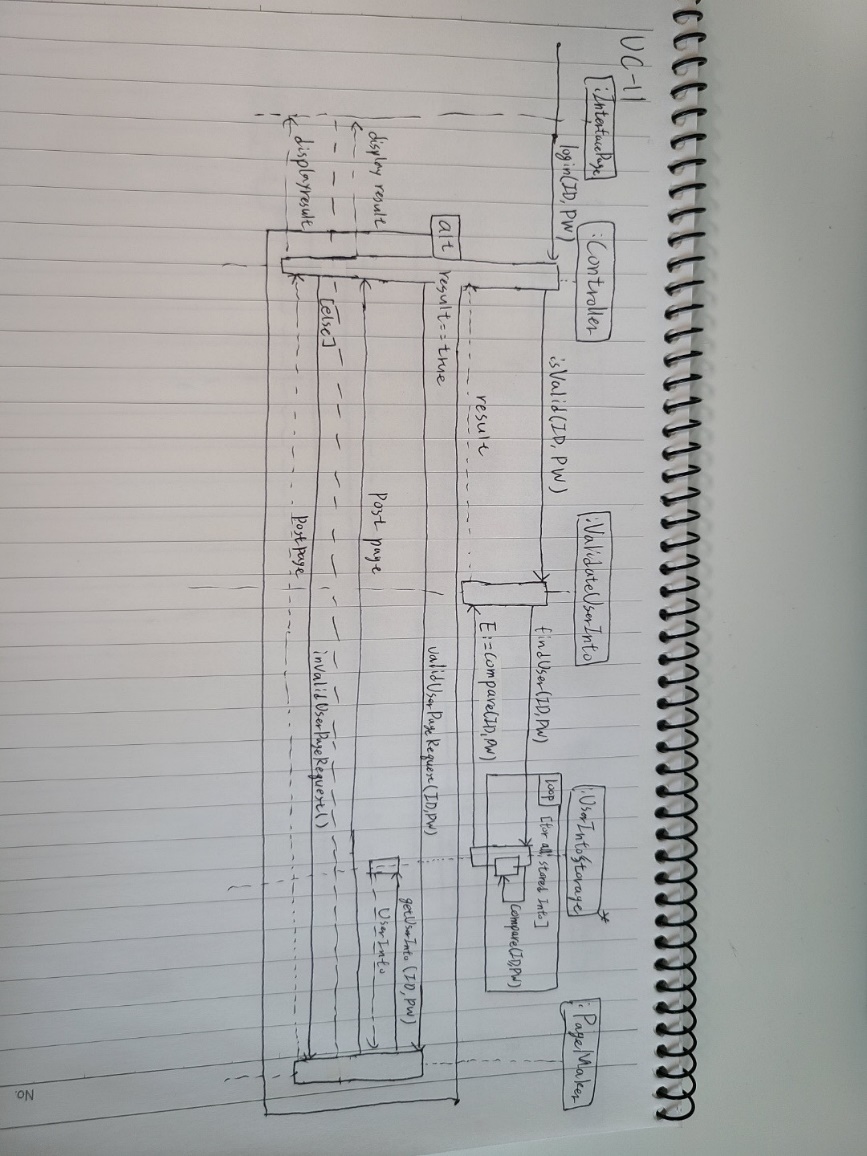


User의 로그인 시도로부터 userID와 userPW를 받아 이를 Controller에 전달한다. Authenticator는 generateUserKey()를 통해 userID와 userPW를 Controller로부터 받아오고, fetchUser()를 통해 Database에서 userID에 대한 userPW를 받아온다.

입력받은 userID, userPW와 Database에서 가져온 userPW를 통해 isValid()를 호출하여 valid한 로그인 시도인지 판단하고, valid할 경우 generateUserKey()로 userKey를 만들고 이를 return해준다.

return이 InterfacePages까지 전달되면, refreshPage()를 통해 로그인 시도 성공 여부를 창에 표시해준다.

1. Variation2

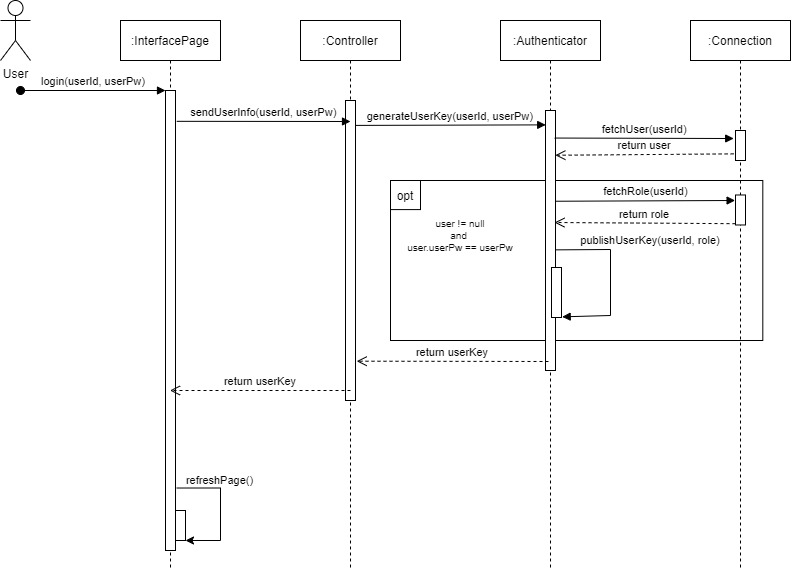


User가 로그인을 하면 Controller가 ID와 PW를 ValidateUserInfo로 전달한다. ValidateUserInfo는 UserInfoStorage에서 전달받은 ID, PW가 있는지 확인하여 로그인 가능 여부를 판단한다.

만약 compare()의 결과가 True라면, PageMaker는 getUserInfo()를 통해 UserInfoStorage에서 해당 유저의 정보를 받아와 페이지를 만들어준다.

False라면 PageMaker는 바로 에러창을 만들어준다.

* + 1. **최종 결과 및 선정 이유**



Variation1을 base로 채택하였다.

일반적인 웹페이지에서 로그인에 성공 또는 실패한 경우를 생각해봤을 때, PageMaker라는 object를 넣어 새로운 페이지를 만드는 것이 불필요하다고 생각되었다.

그리고 Variation2처럼 userID와 userPW를 변환하지 않고 object들이 계속 가지고 있는 것보다 Variation1처럼 userID와 userPW를 userKey로 변환하여 로그인의 valid 여부를 판단하는 것이 domain model을 더 잘 반영한다고 판단하였다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Domain Conecpts | Software class | | | |
| InterfacePage | Controller | Authenticator | Connection |
| InterfacePage | O |  |  |  |
| Controller |  | O |  |  |
| Authenticator |  |  | O |  |
| PageMaker |  |  |  | O |

* 1. **Class Diagram**

