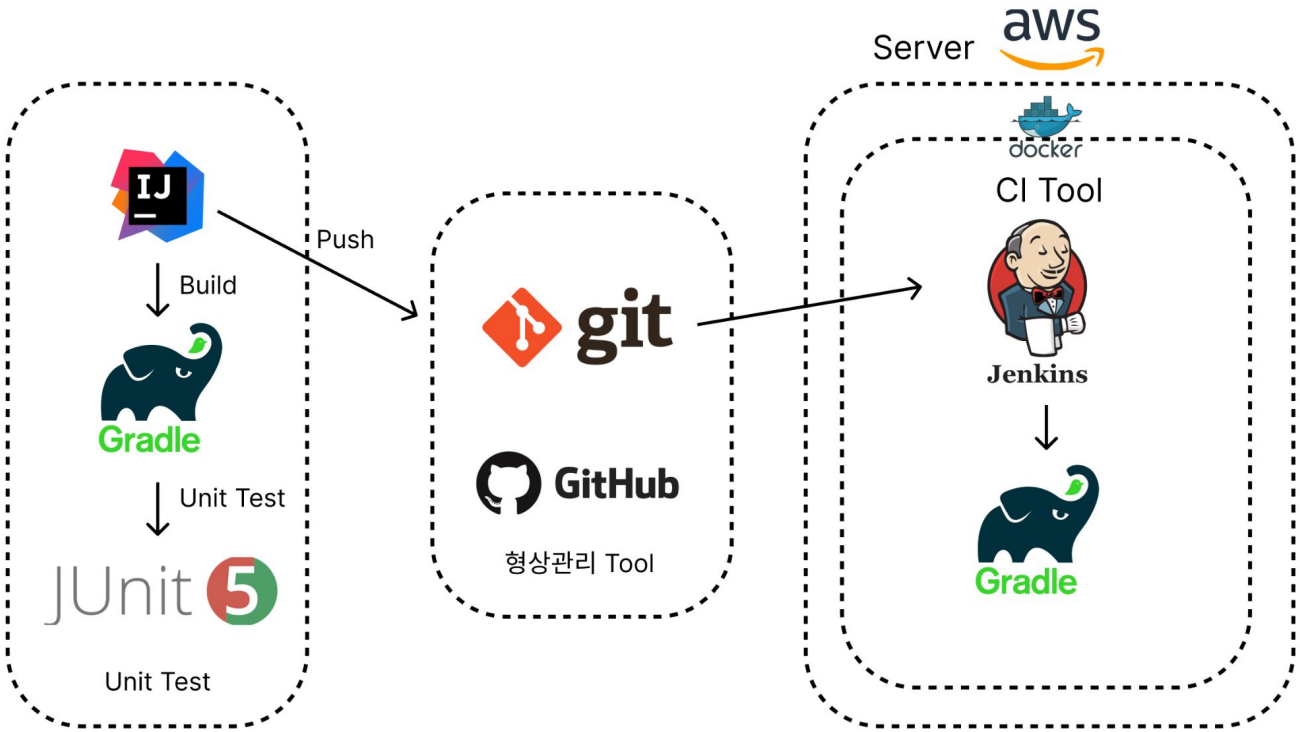
2050 OOI + 최종 발표

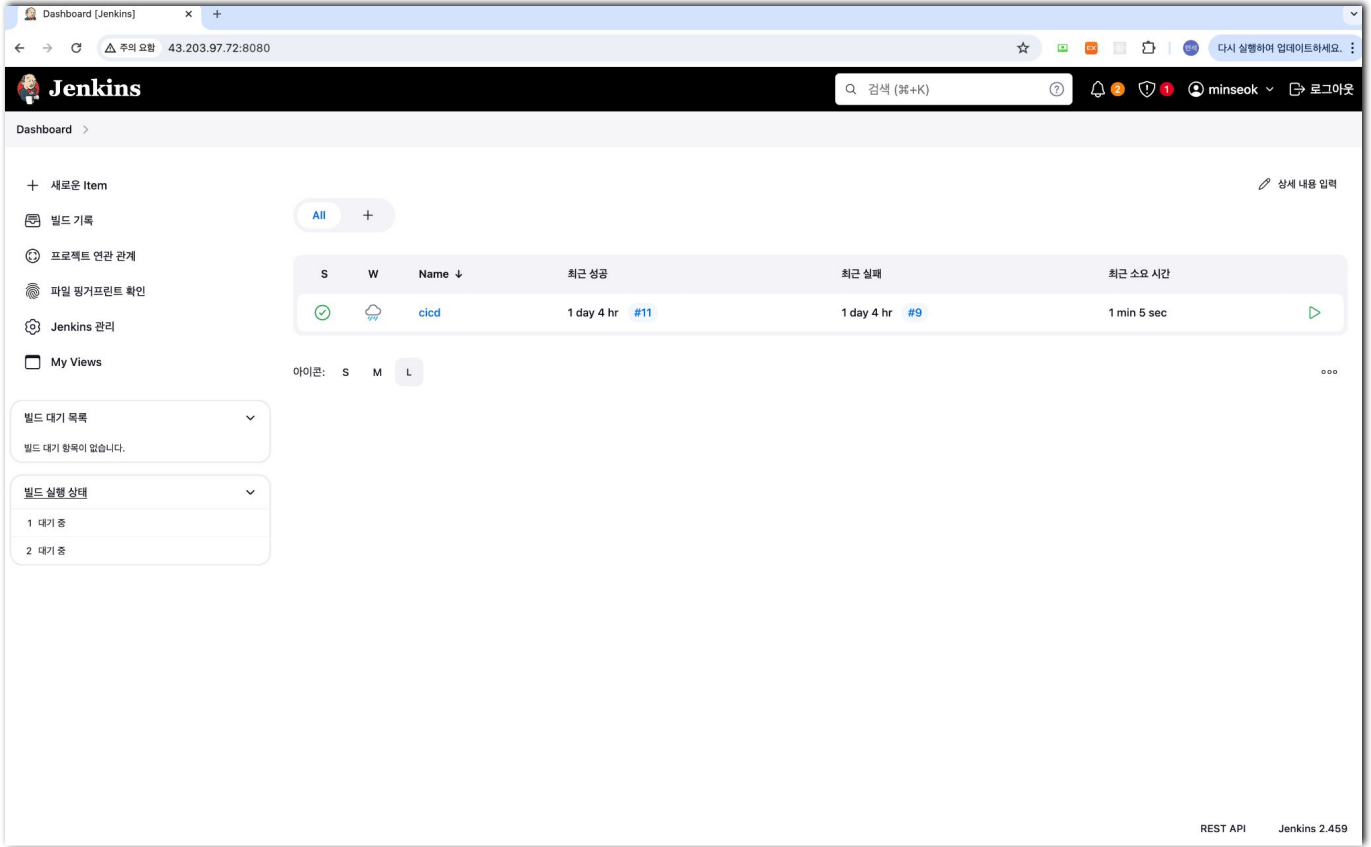
: Distributed vending machine

202211252 고승우
202211300 박연주
202211349 이채유
202214202 김민석

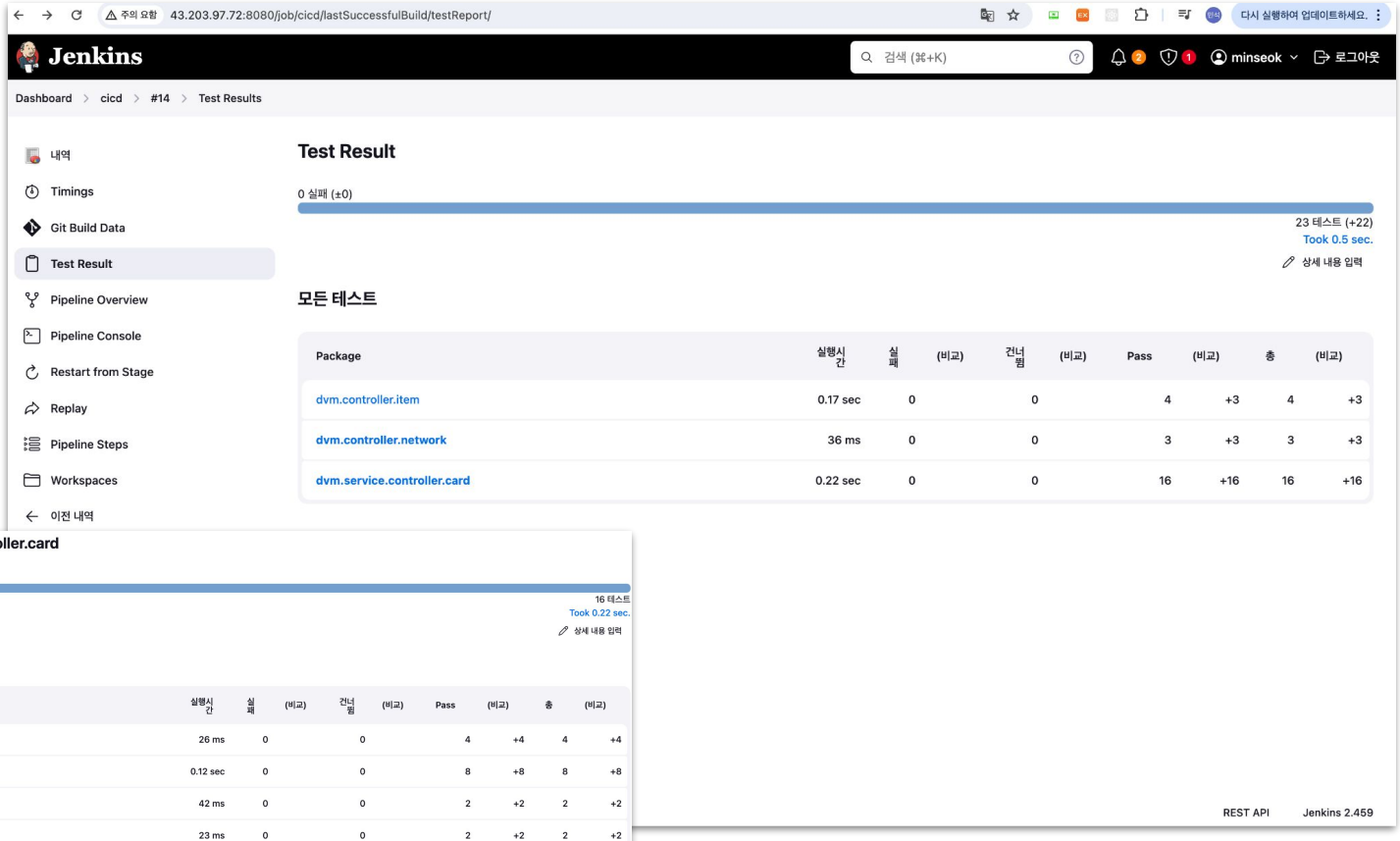
CI/CD


Server
aws
docker
CI Tool
Jenkins
Gradle
Push
Build
Gradle
Unit Test
JUnit 5
Unit Test
git
GitHub
형상관리 Tool

Jenkins Build



Jenkins Test Results



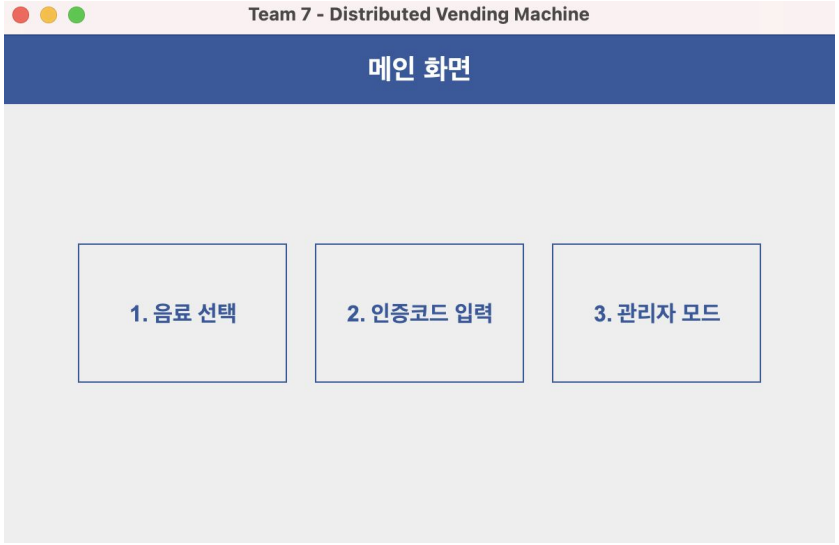
UT 및 System Test 시나리오 결과

Num	Test 항목	Description	Condition	Expect	Pass / Fail
1	PrintMenu	메뉴 20가지가 잘 출력되는지 확인	사용자가 DVM 프로그램을 실행시켰을 때	사용자에게 음료 목록을 제공한다.	Pass
2	Choose Item	선택된 음료 개수와 종류를 확인	사용자가 음료의 종류와 개수를 선택했을 때	사용자가 선택한 음료와 그 개수를 시스템에게 잘 전달한다.	Pass
3	Provide item	음료 개수와 종류가 알맞게 제공됐는지 확인	선결제, 재고확인 등의 DVM이 정상작동할 때	사용자에게 문제없이 음료를 제공하고 화면에 출력한다.	Pass
4	Payment	결제가 정상적으로 진행되어 결제 금액만큼 잔액이 차감됐는지 확인	카드가 유효하고 잔액이 있어야한다.	사용자가 선택한 음료의 가격만큼 카드의 잔액에서 차감된다.	Pass
5	Valid Card	카드가 유효하고 잔액이 있는지 확인	사용자가 카드번호를 올바르게 시스템에게 제공해야한다.	카드번호가 카드사에 있으면 결제할 준비가 된다.	Pass
6	Update Stock	결제 후 음료재고 업데이트 됐는지 확인	결제가 정상적으로 진행된 상태여야 한다.	사용자가 선택한 음료의 개수만큼 재고에서 차감한다.	Pass
7	Update Stock	다른 자판기로부터 선결제 확인 시 재고 차감 확인	선결제 메시지 송수신이 정상적으로 진행된 상태여야 한다.	다른 DVM에서 요청한 음료의 개수만큼 그 음료의 재고에서 차감한다.	Pass
8	Prepayment	선결제 요청 점검	다른 DVM이 사용자가 원하는 음료를 제공할 수 있는지 확인이 되어있어야 한다.	선결제가 가능한지에 대한 응답을 받고 그에 대한 operation을 진행한다	Pass
9	CheckOtherDVMS	우리 DVM에서 재고가 없을 시 다른 DVM과 통신이 되는지 확인	메시지를 송수신할 환경이 마련되어야 하고 사용자가 선택한 음료를 제공할 수 없어야 한다.	음료를 제공할 수 있는 DVM들을 추리고 가장 가까운 DVM을 계산한다.	Pass
10	Provide AuthenticationCode	다른 자판기에 구매 정보 및 인증코드 송신 점검	다른 DVM에게서 선결제가 가능하다는 응답을 받아야 한다.	인증코드를 다른 DVM에게 정상적으로 전달한다.	Pass

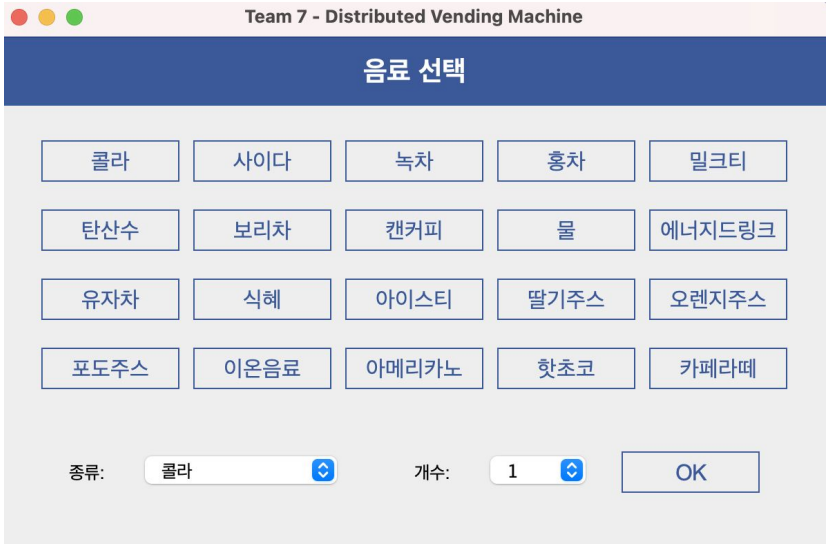
UT 및 System Test 시나리오 결과

Num	Test 항목	Description	Condition	Expect	Pass / Fail
11	Provide Authentication Code	선결제 시 인증코드가 잘 발급되는지 확인	다른 DVM에게서 선결제가 가능하다는 응답을 받아야 한다.	DVM이 생성한 인증코드를 사용자에게 정상적으로 출력한다.	Pass
12	Provide Available DVM Location	선결제된 DVM의 위치를 사용자에게 알맞게 주었는지 확인	선결제 요청에 대한 응답을 받을 때 DVM의 위치가 담긴 메시지를 정상적으로 송신 받아야한다.	선결제가 된 다른 DVM의 위치를 정확히 출력한다.	Pass
13	Check Authentication Code	입력한 인증코드가 다른 DVM이 발급한 인증코드와 같은지 확인	메시지를 수신을 통해 다른 DVM에서 생성한 인증코드를 받는다.	다른 DVM에서 보낸 인증코드와 같으면 음료를 제공하는 operation을 실행한다.	Pass
14	Request From Other DVM	다른 DVM에서 stock에 대한 request가 올 때 정상적으로 통신이 되는지 확인	서버가 열려있어야 한다.	다른 DVM에서 보낸 메시지가 정상적으로 수신된다.	Pass
15	Admin Login	관리자모드로 접속 되는지 확인	프로그램이 실행되는 상태여야 한다.	알맞게 관리자 ID와 Password를 입력해 관리자 모드로 진입한다.	Pass
16	Admin Logout	관리자모드 로그아웃 기능 확인	관리자모드로 진입된 상태여야 한다.	메인화면으로 돌아간다.	Pass
17	Set Item Stock	관리자모드, 음료 재고 조정 확인	관리자모드로 진입된 상태여야 한다.	관리자가 입력한 음료의 재고 수정 값이 DVM ItemRepository에 반영된다.	Pass
18	Set Item Price	관리자모드, 음료 가격 조정 확인	관리자모드로 진입된 상태여야 한다.	관리자가 입력한 음료의 가격 수정 값이 DVM ItemRepository에 반영된다.	Pass

시스템 동작 Demo[1. 음료 선택]

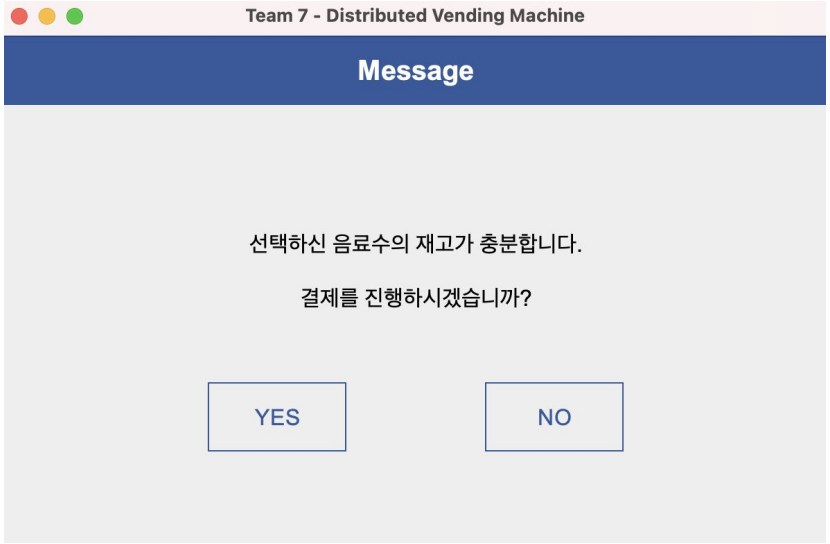


시스템 초기 화면



[1. 음료 선택] 버튼 선택 시

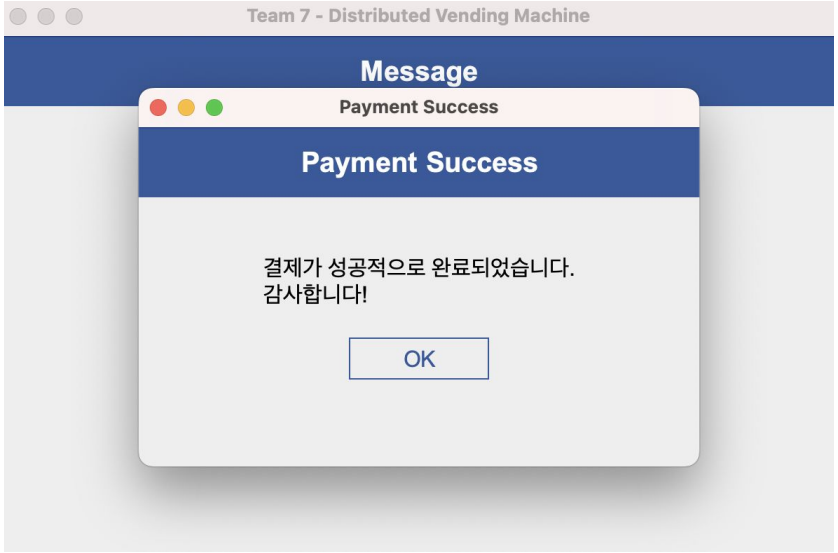
시스템 동작 Demo [1. 음료 선택] - DVM에 재고가 있는 경우



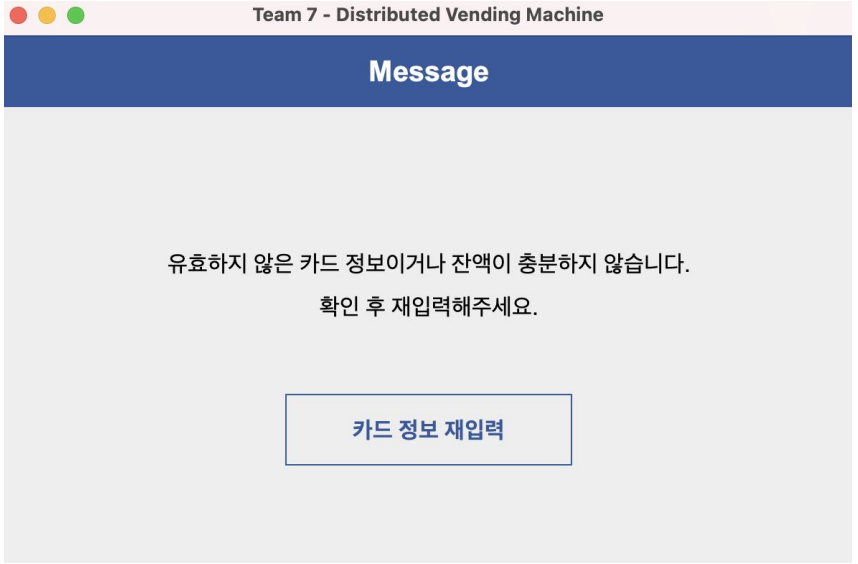
원하는 음료 선택 시,
결제 진행 여부

YES -> 카드 정보 입력
NO -> 메인 화면으로 돌아감

시스템 동작 Demo [1. 음료 선택] - DVM에 재고가 있는 경우

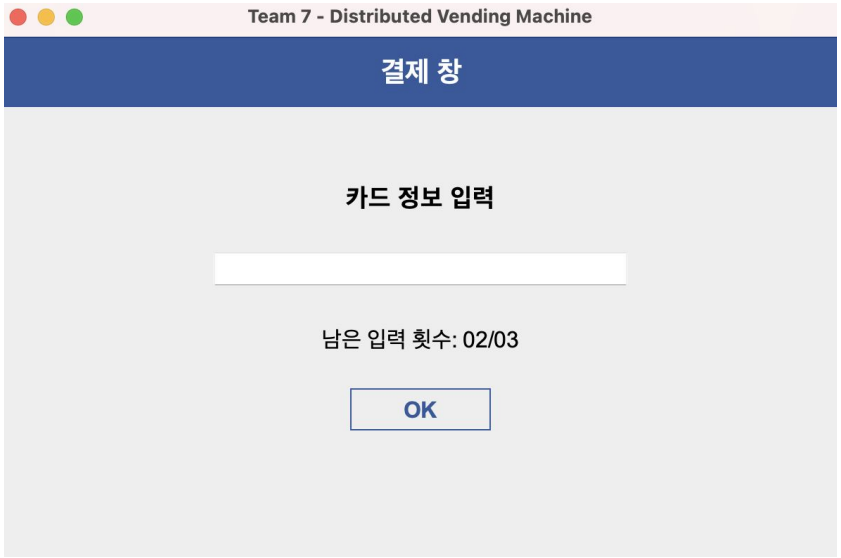


올바른 카드 번호 입력 및 잔액 충분할 경우
: OK 버튼 선택 시, 메인화면으로 돌아감

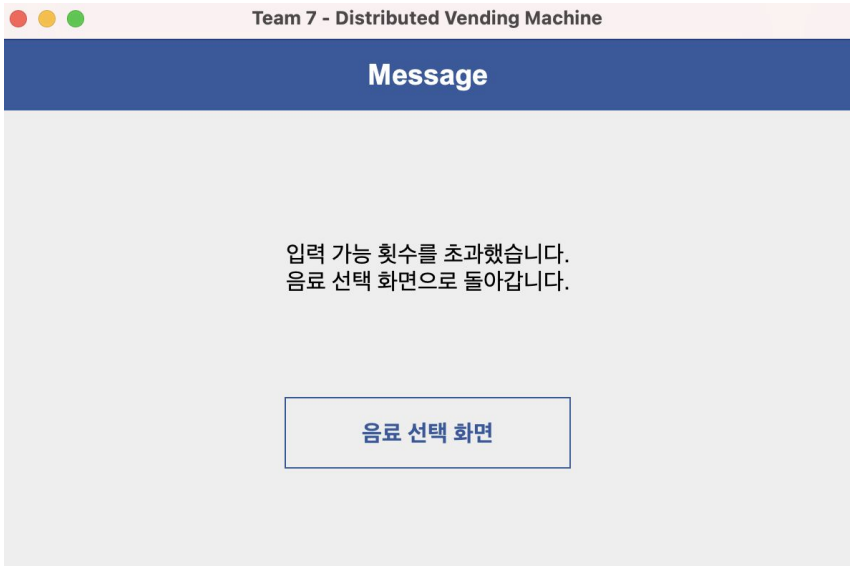


올바르지 않은 카드 번호 입력 혹은 잔액이 충분하지 않을 경우
: 카드 정보 재입력 버튼 누를 시, 결제화면 재출력

시스템 동작 Demo [1. 음료 선택] - 다른 DVM에 재고가 있는 경우

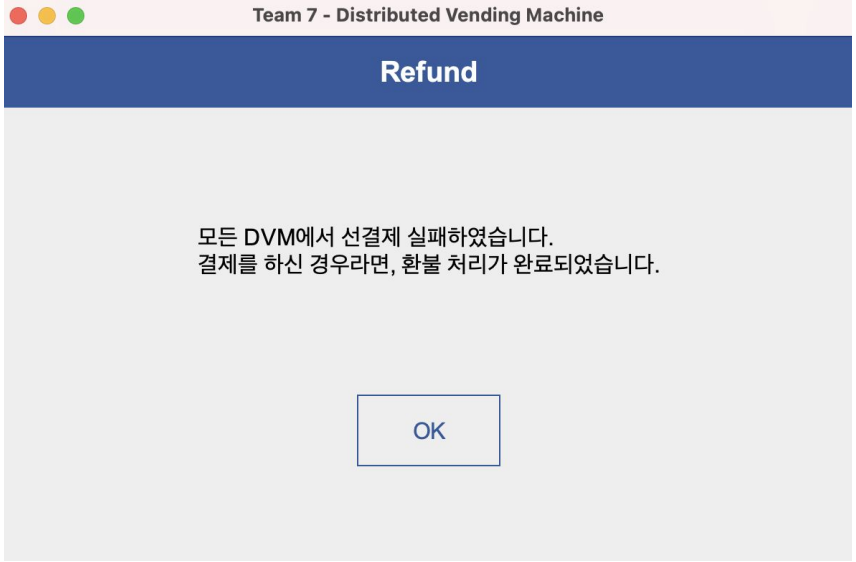


카드 결제 실패 시, 횟수 차감 및 결제창 재출력



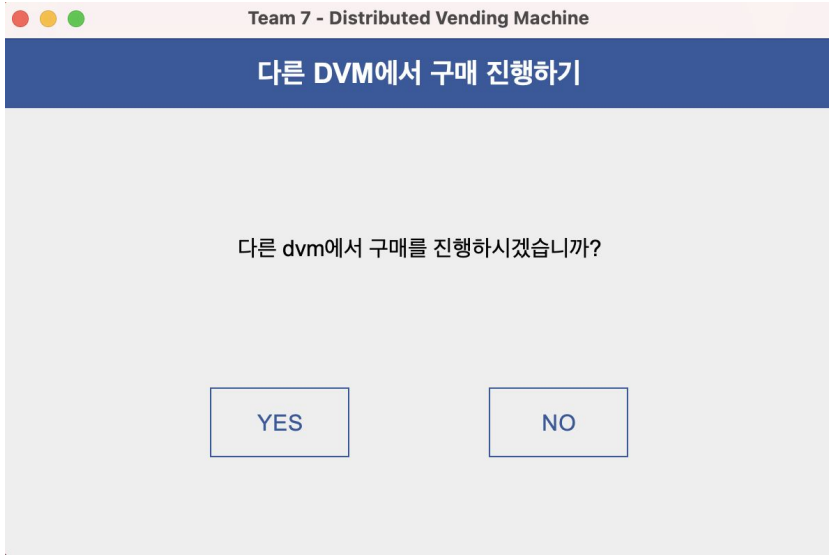
입력 가능 횟수 초과 시, 메세지 출력 후 버튼 누르면 음료 선택 화면 출력

시스템 동작 Demo [1. 음료 선택] - DVM에 재고가 없는 경우



재고가 있는 다른 DVM이 존재하지 않을 시

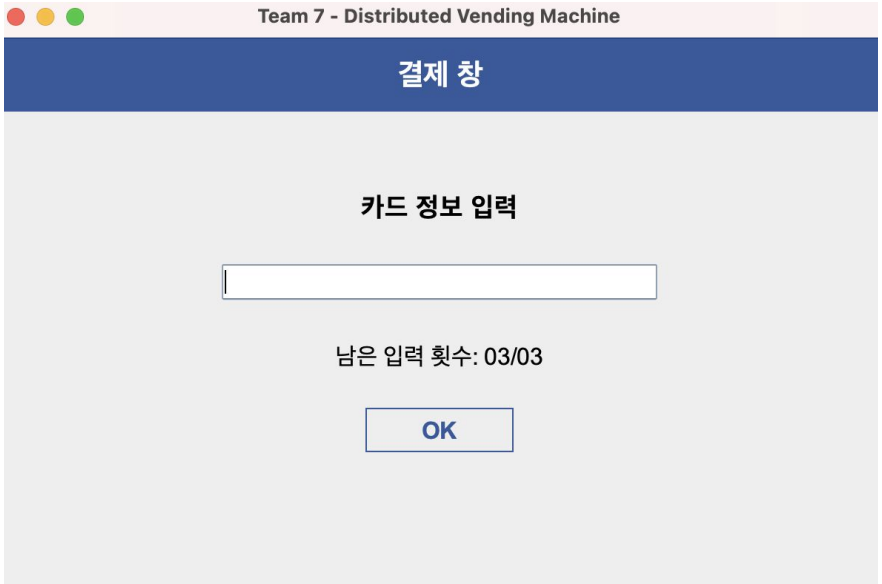
: OK 버튼 누르면, 메인 화면으로 돌아감



재고가 있는 다른 DVM이 존재하는 경우

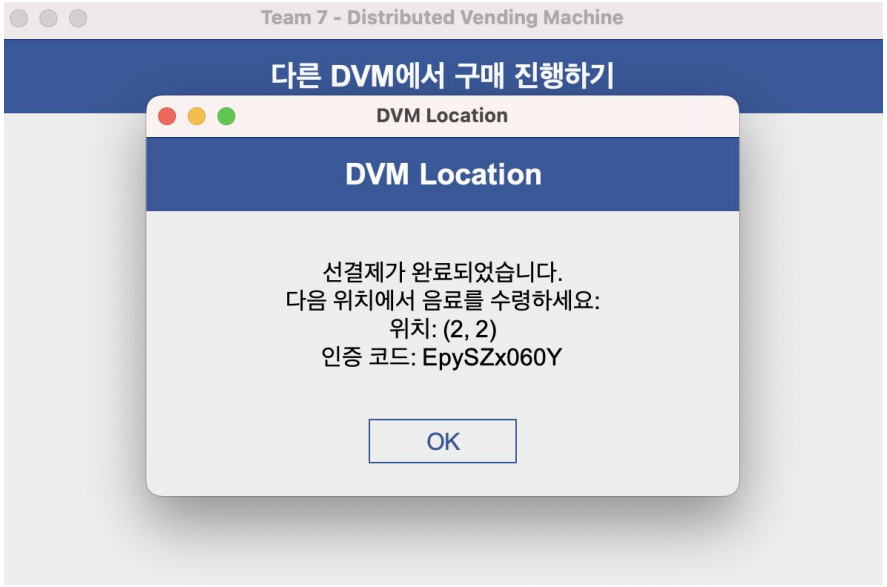
: YES -> 선결제를 위한 카드창 출력
NO -> 메인 메뉴창 출력

시스템 동작 Demo [1. 음료 선택] - 다른 DVM에 재고가 있는 경우



선결제 위한 카드 결제창

: 결제 실패의 경우, 해당 DVM에서 결제 실패한 경우와
동일하게 출력

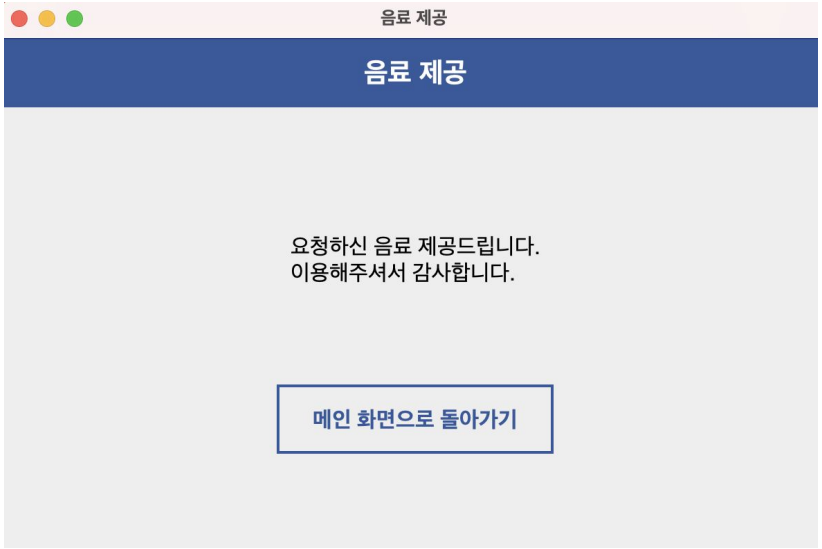


선결제 성공 시

: 사용자에게 정보 제공 팝업창 뜨고 버튼 누르면
메인화면 출력

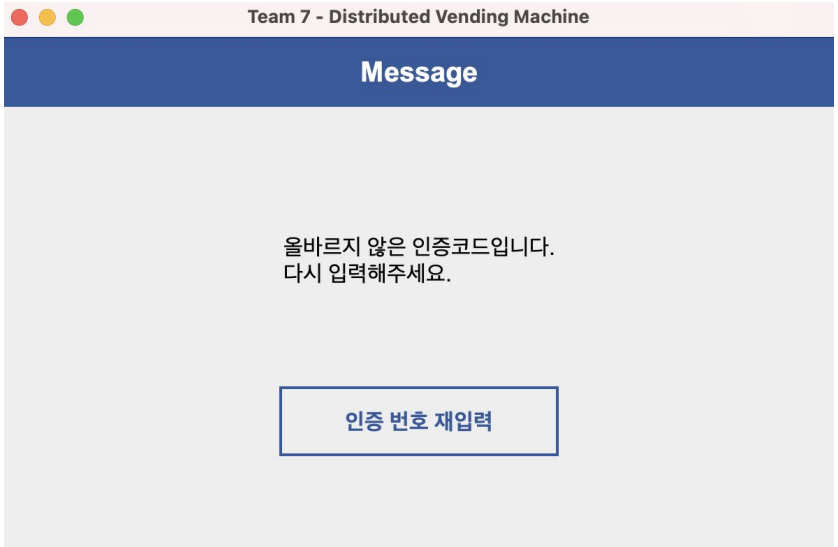
시스템 동작 Demo [2. 인증코드 입력]

[2. 인증코드 입력] 버튼 선택 시



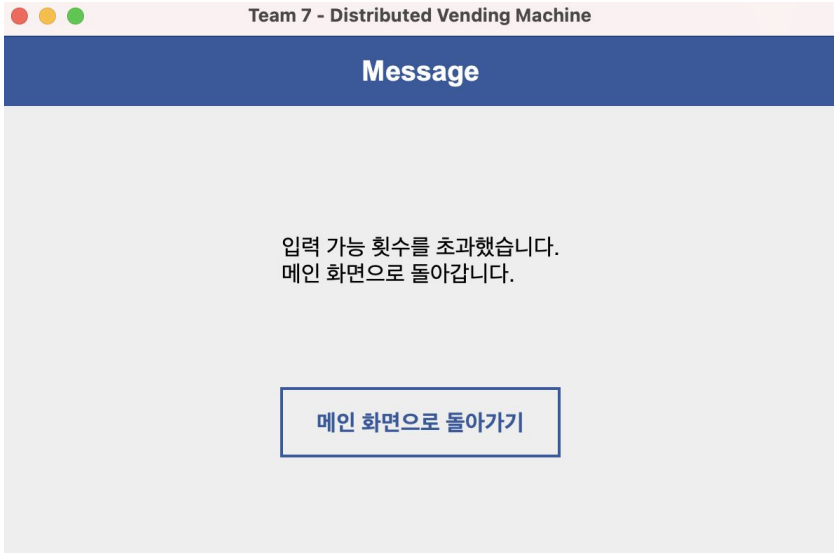
올바른 인증코드 입력 시

시스템 동작 Demo [2. 인증코드 입력]



올바르지 않은 인증코드 입력 시

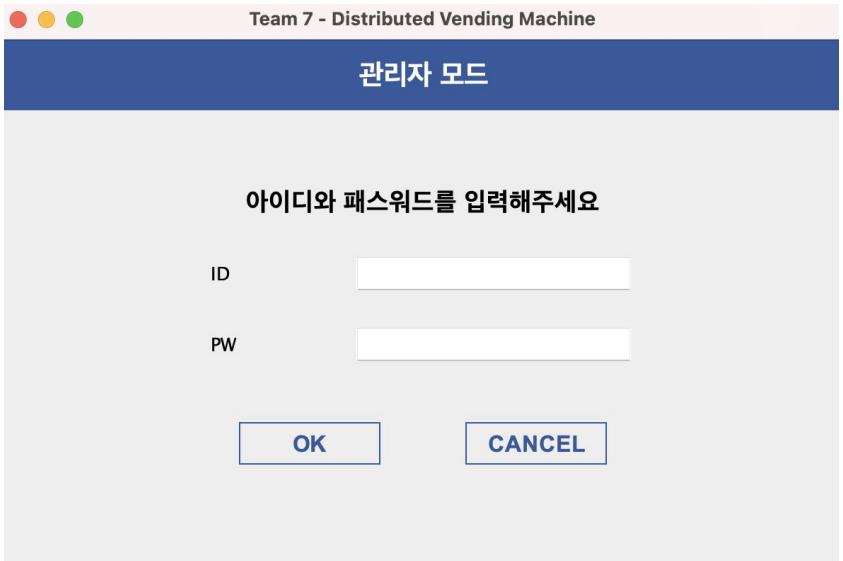
: 횟수 차감 후, 인증코드 입력 창 재출력



입력 횟수 초과 시

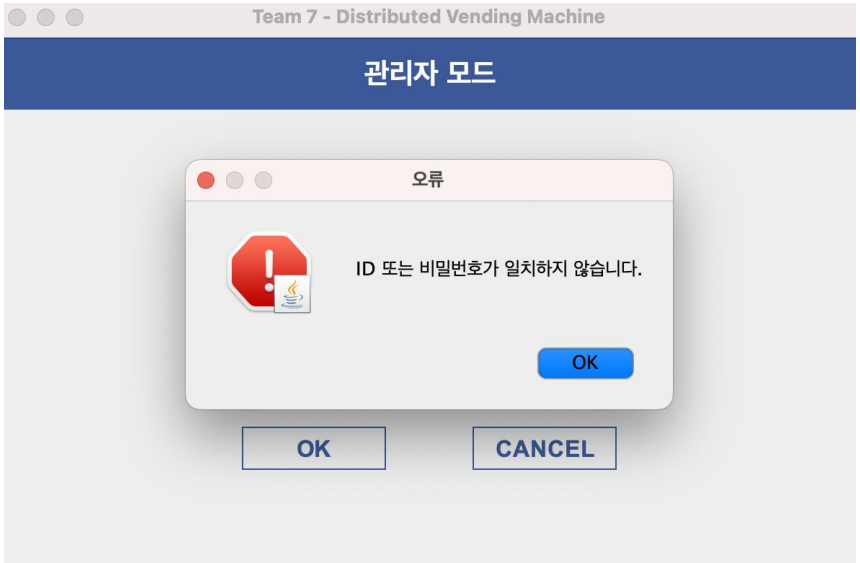
: 메인 화면으로 돌아감

시스템 동작 Demo [3. 관리자 모드]



[3. 관리자 모드] 버튼 선택 시

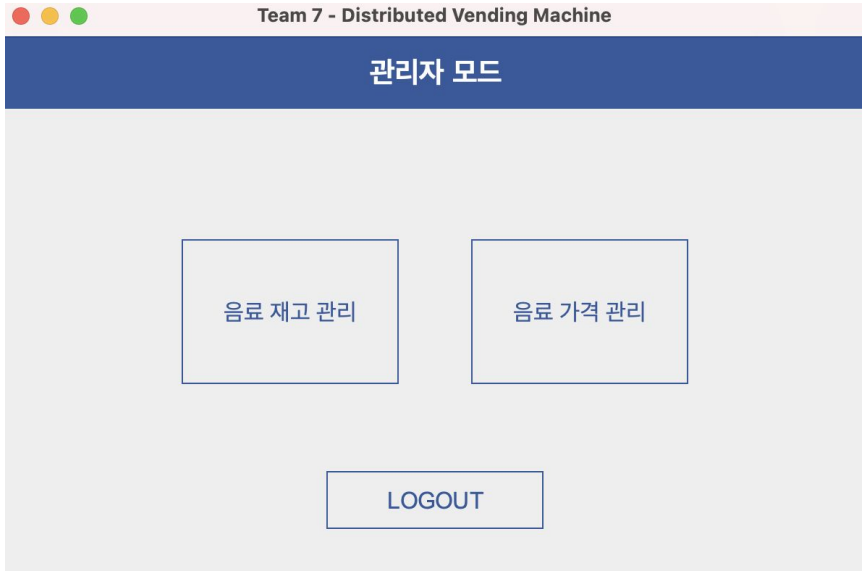
: [CANCEL] 버튼 선택 시, 메인화면으로 돌아감



로그인 실패

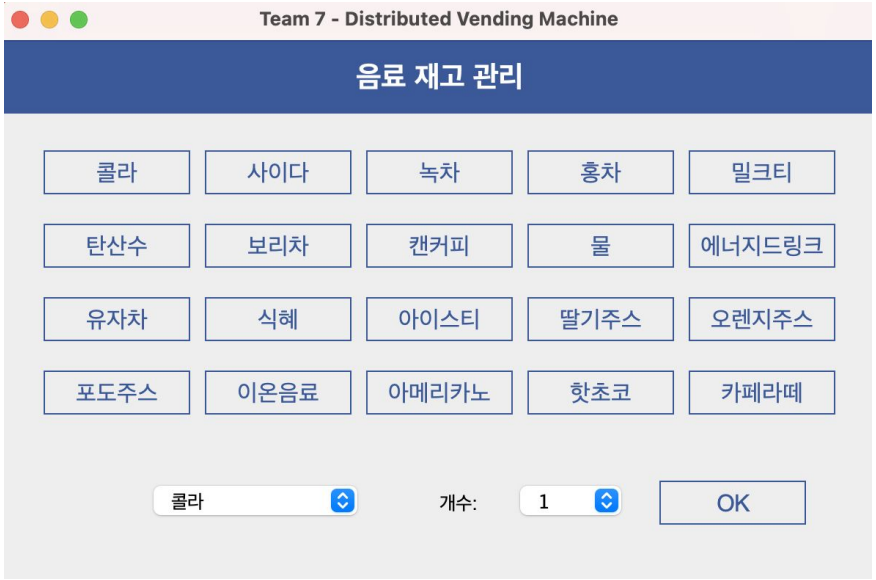
: OK 버튼 선택 후, 로그인 화면 재출력

시스템 동작 Demo [3. 관리자 모드]



로그인 성공 시 관리자 모드 메뉴 출력

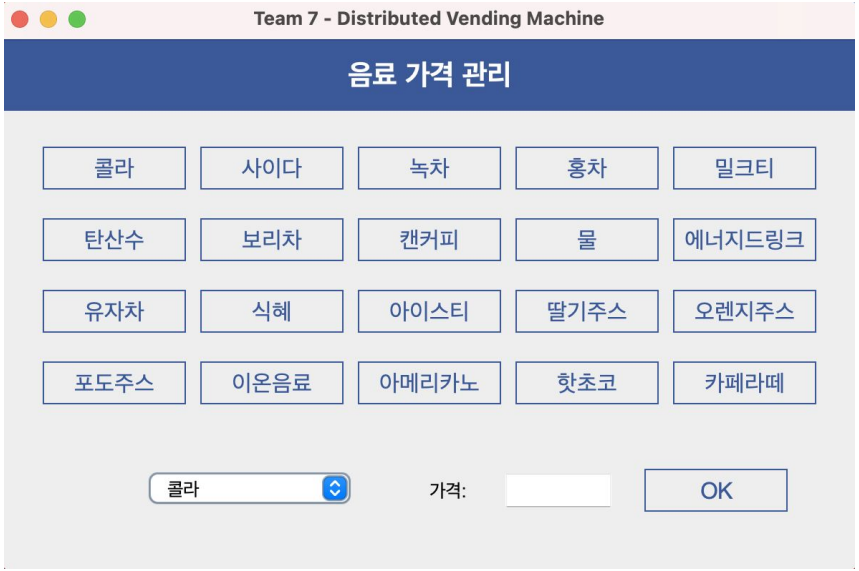
+) LOGOUT 버튼 선택 시, 메인화면 출력



[음료 재고 관리] 선택 시

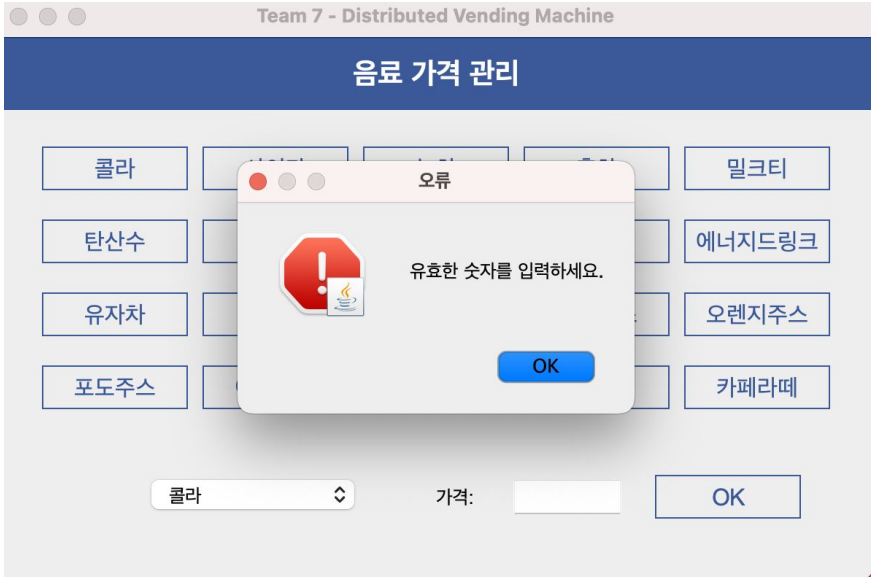
+) 원하는 음료 및 개수 선택 후, OK 버튼 누르면 관리자 모드 메뉴 출력

시스템 동작 Demo [3. 관리자 모드]



[음료 가격 관리] 선택 시

+) 원하는 음료 선택 및 가격 작성 후,
OK 버튼 누르면 관리자 모드 메뉴 출력

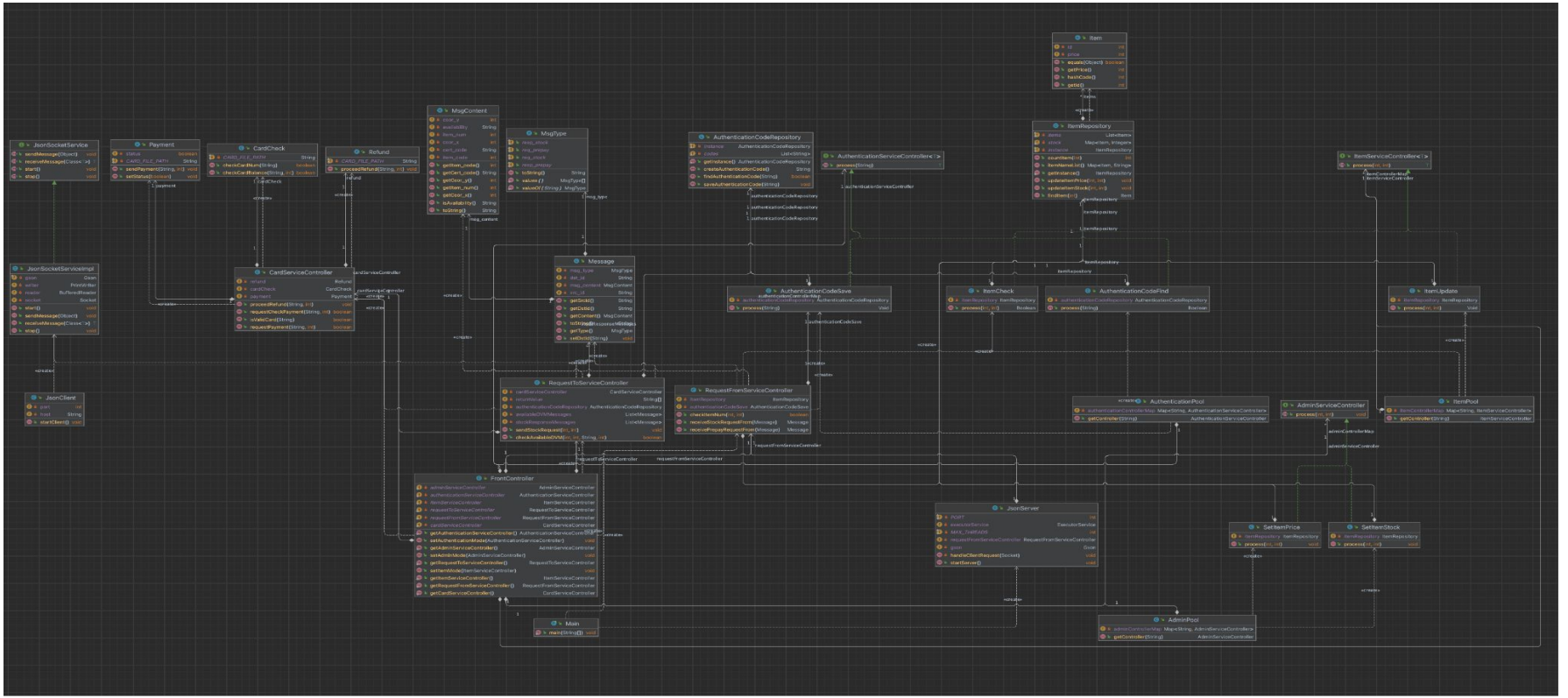


유효하지 않은 가격 입력 시

: 숫자가 아닌 문자 입력 시 오류 발생
+) OK 버튼 누르면 음료 가격 관리 창 재출력

OOD(2040)에서 변경된 부분

클래스 다이어그램



OOD(2040)에서 변경된 부분

1. 패키지 & 클래스 구조 변경
   - FrontController class의 역할을 UI패키지의 각 class가 맡게 되었다.
   - network 패키지 하위에 JsonSocketService interface를 추가
   - 이를 구현한 JsonSocketServiceImpl class 생성
   - 실질적 Server역할을 하는 JsonServer class 추가
   - RequestFromServiceController class는 수신한 요청 메시지를 처리해 응답 메시지를 반환하는 역할로 분리

2. 메소드 추가 및 변경
   - RequestFromServiceController class의 sendStockReqeustFrom, sendPrepayRequestFrom 메소드는 동일 class에 같은 기능을 하는 메소드가 존재해 삭제

구현 시 예상보다 어려웠던 점

- CTIP 환경을 처음 사용해봐서 초기 환경 구축에 있어서 어려움을 겪었다.

- 소켓 통신 환경을 사용해보지 않아서 테스트를 해보고 다른 팀의 DVM과 통신이 제대로 되는지 확인하기 어려운 부분이 있었다.

- sequence diagram과 class diagram에서 일관되지 않은 부분이 발견되었을 때, 오류를 겪어 2040 stage의 중요성을 느낄 수 있었다.

구현 시 예상보다 쉬웠던 점

- 구현해야하는 사항들을 요구사항 분석 단계에서 미리 다 use case로 작성해놔서 개발 단계에서 편리하였다.

- 기능별로 class들을 미리 구분해서 설계해놔서 구현 시, class별로 구현할 때 생각보다 간단하였다.

- 테스트 케이스 작성 후 기능 별 테스트를 할 때, CTIP 환경을 구축해놔서 빠르고 간편하게 테스트를 확인할 수 있었다.

- 코드를 작성할 때, sequence diagram을 보고 작성하면 돼서 핵심 로직을 간편하게 작성할 수 있었다.

객체지향개발방법론의 장단점

장점:

1. 설계 단계에서 많은 시간이 소요되지 않음
2. 문서화 단계에서 프로젝트에 대해 요구사항 등 구체적인 이해 가능
3. 객체지향개발의 원리를 잘 지켜서 설계 및 구현할 시, 유지보수에 용이

단점:

1. 중간에 수정사항이 생기면 초기 단계부터 다 수정해야하는 번거로움이 존재
   -> 초기 설계 단계에서의 어려움
2. 객체지향개발방법론에서의 원칙 (SRR, OCP, LSP, ISP, DIP 등)을 지키지 않으면 이후 유지보수 단계에 있어서 어려움 존재
3. 해당 방법론에 익숙치 않을 경우 오랜 시간과 노력이 필요

개인적인 소감

김민석 : UseCase들의 내용을 가지고 Sequence Diagram을 '잘' 짜려는게 생각보다 어려웠다. Sequence Diagram을 잘 짜려면 많은 고민이 필요했던 것 같다.

박연주 : CTIP 환경 구축과 소켓 통신이 처음이라 어려움을 많이 겪었다. OOPT 적용은 문서작업 및 설계 단계에는 시간이 많이 들지만, 깔끔하고 효율적인 프로그램 구현이 가능한 것 같다.

개인적인 소감

고승우 :개발의 모든 단계를 단순히 이론에서 배우는 것이 아닌 몸으로 경험할 수 있어서 좋았습니다. 여러 UML도구를 써보며 설계를 하고 설계한 것을 구현하는 것까지 이전에 해왔던 방식과는 달라서 좋았습니다.

이채유 : 설계 단계에서 CTIP, 소켓 통신 구축 등 다뤄보지 않은 것들이 존재하여 생각보다 어려움을 겪었다. 하지만 OOPT 1000 - Design 단계에서의 요구사항 분석부터 시작하여 2050 OOI 단계까지 진행하면서 객체지향개발방법론에 대해 배울 수 있었다. 아직 요구사항 분석 및 sequence diagram 작성 등 익숙치 않은 부분이 많아 오래 걸리긴 하였지만 더 공부하며 적용해나간다면 대규모 프로젝트에서 잘 사용할 것 같다. 생각보다 설계와 구현 단계 사이에 고민해야할 부분이 많다는 것을 느꼈다.