Final Presentation

& OOPT 2050 OOI

Team 01

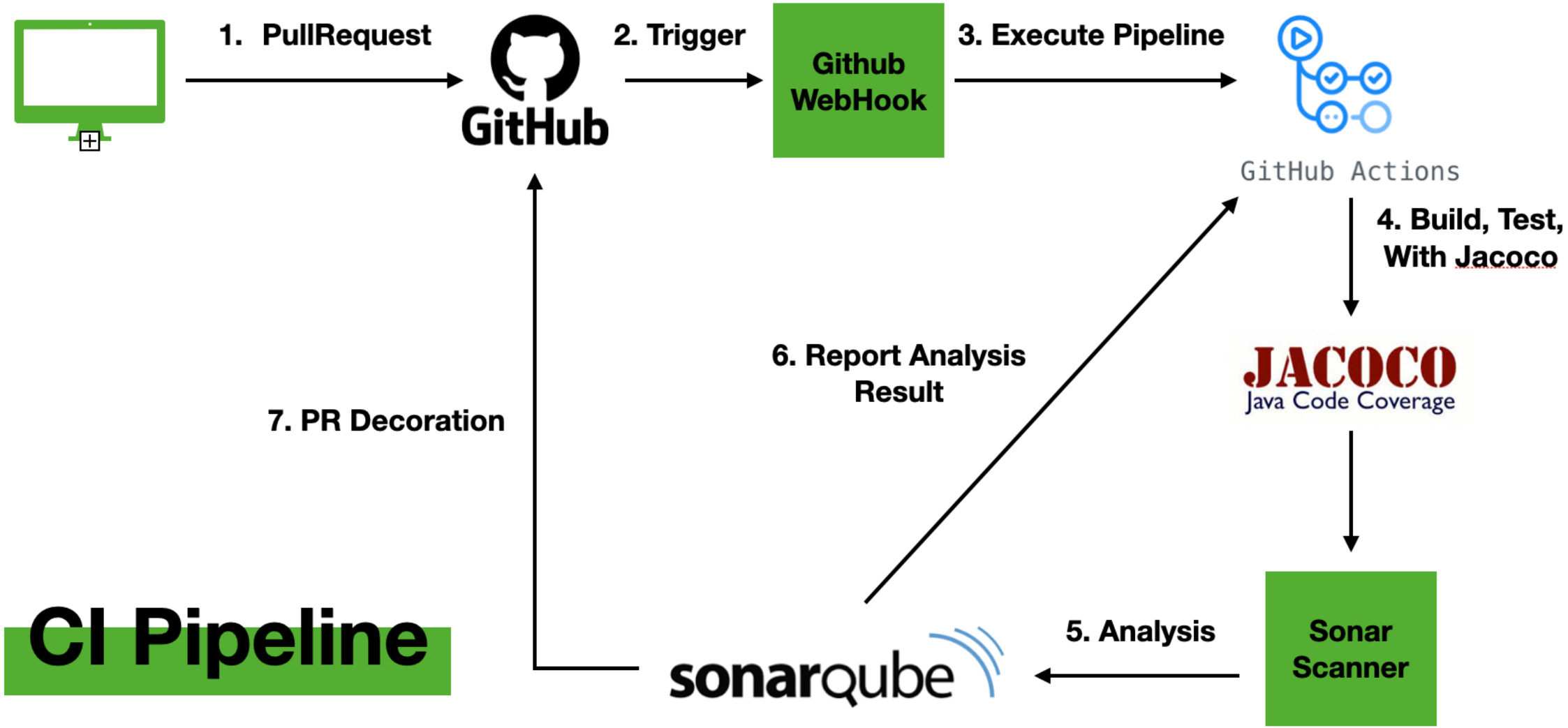
2018142 63 김지우
2019111 85 신동성
2021123 40 이병직
2019111 93 우이산

목차

Table of Contents

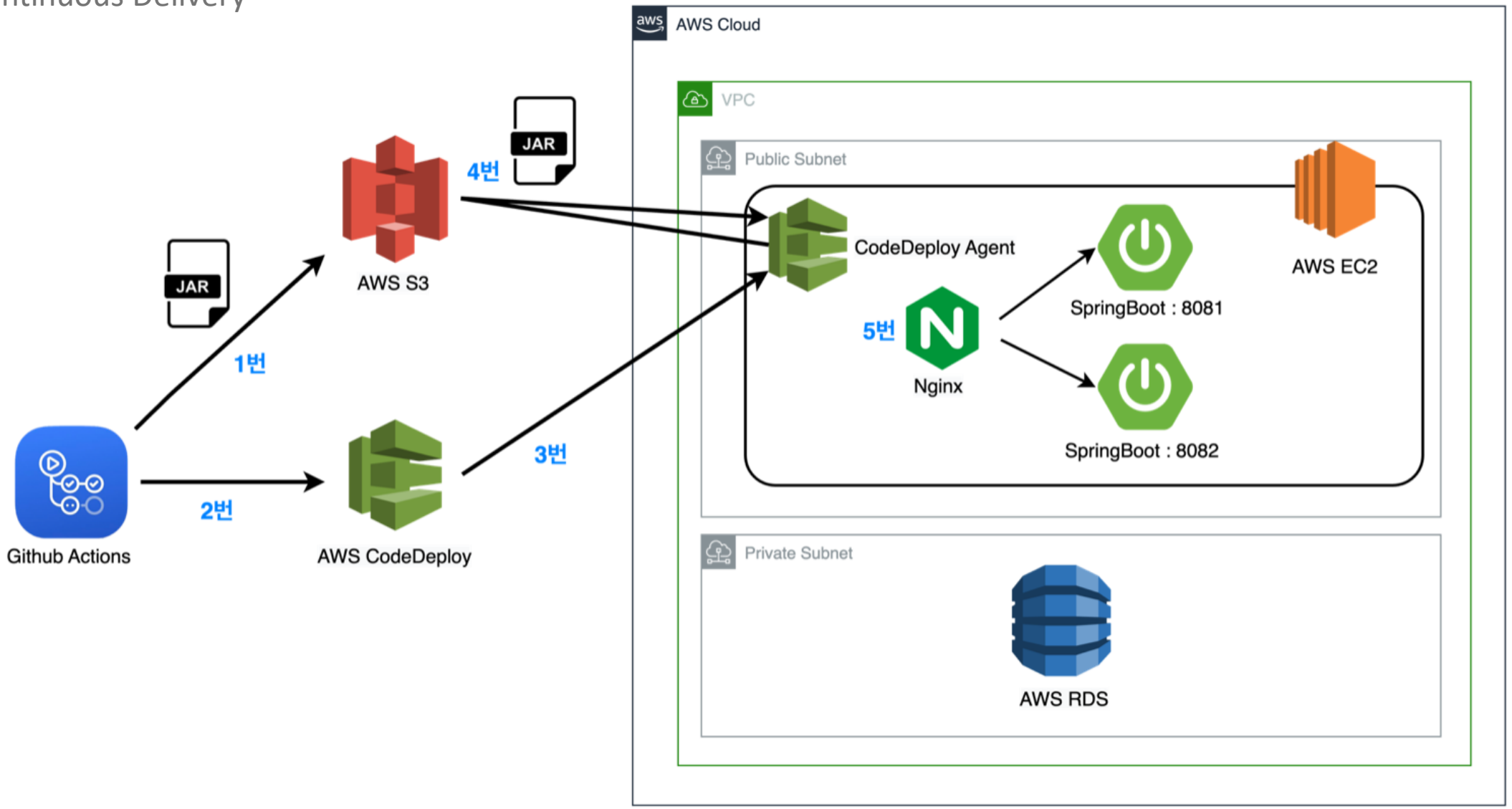
1. 개발환경(CI/CD)

Continuous Integration



1. 개발환경(CI/CD)

Continuous Delivery



2. UT 및 System Test 시나리오 및 결과

Num	Use Case	Description	Given	When	Then	Pass / Fail
1	1. 구매 요청	사용자가 구매 가능한 음료 목록을 볼 수 있어야 한다.	사용자가 DVM에 접근했을 때	음료 메뉴를 요청하면	시스템은 사용자에게 구매 가능한 음료 목록을 제공한다.	Pass
2	1. 구매 요청	사용자는 선택한 음료와 수량을 입력하여 구매 요청을 보낼 수 있어야 한다.	사용자가 특정 음료와 수량을 선택했을 때	사용자가 선택한 음료와 수량에 대한 구매 요청을 보내면	시스템은 사용자의 요청을 처리하고 구매 진행을 위한 다음 단계로 넘어간다.	Pass
3	2. 재고 확인	현재 기기에서 요청한 수량 이상을 가지고 있는지 확인	사용자가 희망 음료와 수량을 요청	현 기기에 재고가 충분	즉시 결제 시도	Pass
4	2. 재고 확인	요청한 수량을 가지고 있는 다른 기기 파악	사용자가 희망 음료와 수량을 요청	현 기기에 재고가 부족	가까운 기기로 안내또는 결제 실패	Pass
5	3. 즉시 결제	현재 기기에 수량이 충분하여 결제 진행	현재 기기에서 결제 시도	사용자가 결제 희망	Bank에 결제를 요청하고 결과 알림	Pass
6	3. 즉시 결제	결제가 성공적으로 진행되었을 경우 음료를 제공	현재 기기에서 결제 시도	결제 성공	음료를 제공하고 재고 갱신	Pass
7	4. 선결제	사용자의 선결제 여부 판단	현재 DVM에 음료가 존재하지 않고 다른 DVM에서 구매 가능할 때	시스템이 사용자에게 선결제를 원하는지 물었을 때	사용자는 선결제 의사를 시스템에 명확하게 전달할 수 있어야 한다.	Pass
8	4. 선결제	사용자가 선결제를 원하면 결재 정보를 입력하여 결재를 진행한다.	사용자가 선결제를 원한다고 응답했을 때	사용자가 계좌 정보를 입력하고, 은행 시스템이 계좌 잔액을 확인한 결과 충분한 잔액이 있을 때	은행 시스템은 결제를 성공적으로 진행한다.	Pass

2. UT 및 System Test 시나리오 및 결과

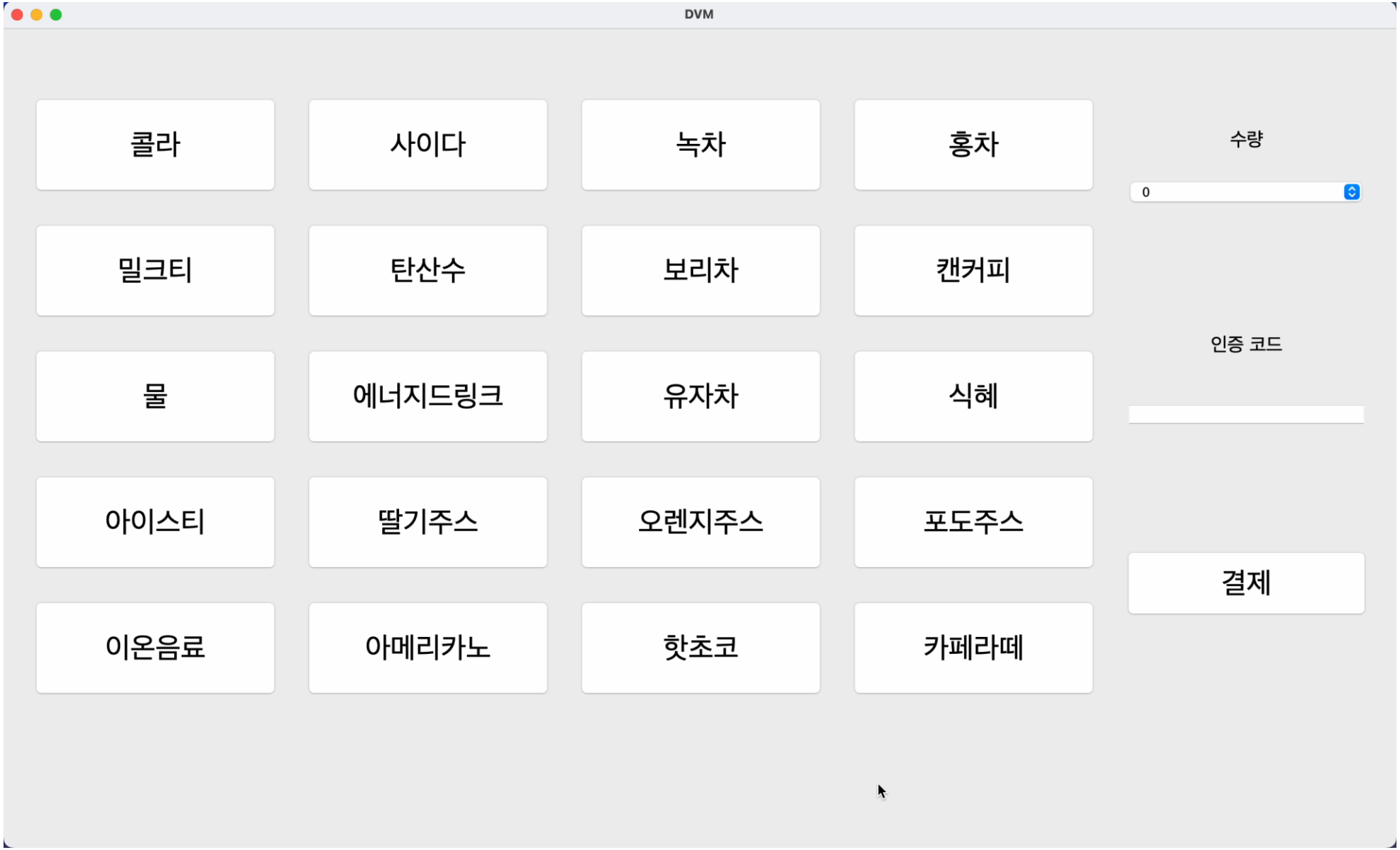
Num	Use Case	Description	Given	When	Then	Pass / Fail
9	5. 선결제 요청	선결제가 정상적으로 수행되는 지 확인한다	선결제를 통하여 사용자 계좌에서 금액이 이미 차감된 상태	선결제 요청시	성공적으로 인증코드를 반환	Pass
10	5. 선결제 요청	선결제 요청 실패	선결제를 통하여 사용자 계좌에서 금액이 이미 차감된 상태	선결제 요청시	예외 발생, 결제 취소 로직 수행	Pass
11	6. 선결제 요청 처리	선결제 물품의 판매 가능 여부 판단	선결제 물품 재고가 없을 때	선결제 요청 처리를 시작하면	실패한다	Pass
12	6. 선결제 요청 처리	선결제 요청 처리	선결제 물품 재고가 있을 때	선결제 요청 처리를 시작하면	선결제 물품을 판매처리하고 인증코드를 등록한다	Pass
13	7. 재고 확인 응답	구매 요청 물품 재고가 충분한 상황이면 재고 충분을 응답	재고가 있을 때,	재고 확인 요청을 받으면	재고 충분과, 현 기기의 위치를 응답	Pass
14	7. 재고 확인 응답	구매 요청 물품의 재고가 불충분한 상황이면 재고 부족을 응답	재고가 없을 때,	재고 확인 요청을 받으면	재고 부족과, 현 기기의 위치를 응답	Pass
15	8. 선결제 음료 제공	인증코드가 유효한 경우 구매 물품 제공	사용자가 제시할 인증코드가 등록되어 있을 때	인증코드를 제시하면	선결제 물품을 제공	Pass
16	8. 선결제 음료 제공	인증코드가 유효하지 않은 경우 물품을 제공하지 않아야 함	사용자가 제시할 인증코드가 등록되어 있지 않을 때	인증코드를 제시하면	선결제 물품을 반환하지 않아야 함	Pass
17	9. 결제 취소	결제 취소에 성공한다	선결제 또는 즉시 결제중인 상황	결제 취소 요청을 하면	성공적으로 결제를 취소한다.	Pass
18	9. 결제 취소	결제 취소에 실패	결제 상황이 아니거나, 사전에 지출된 금액이 없는 경우	결제 취소 요청을 하면	결제가 실패하고, 이를 로그로 남긴다.	Pass

3. 시스템 동작 Demo 영상

즉시결제요청
및 성공

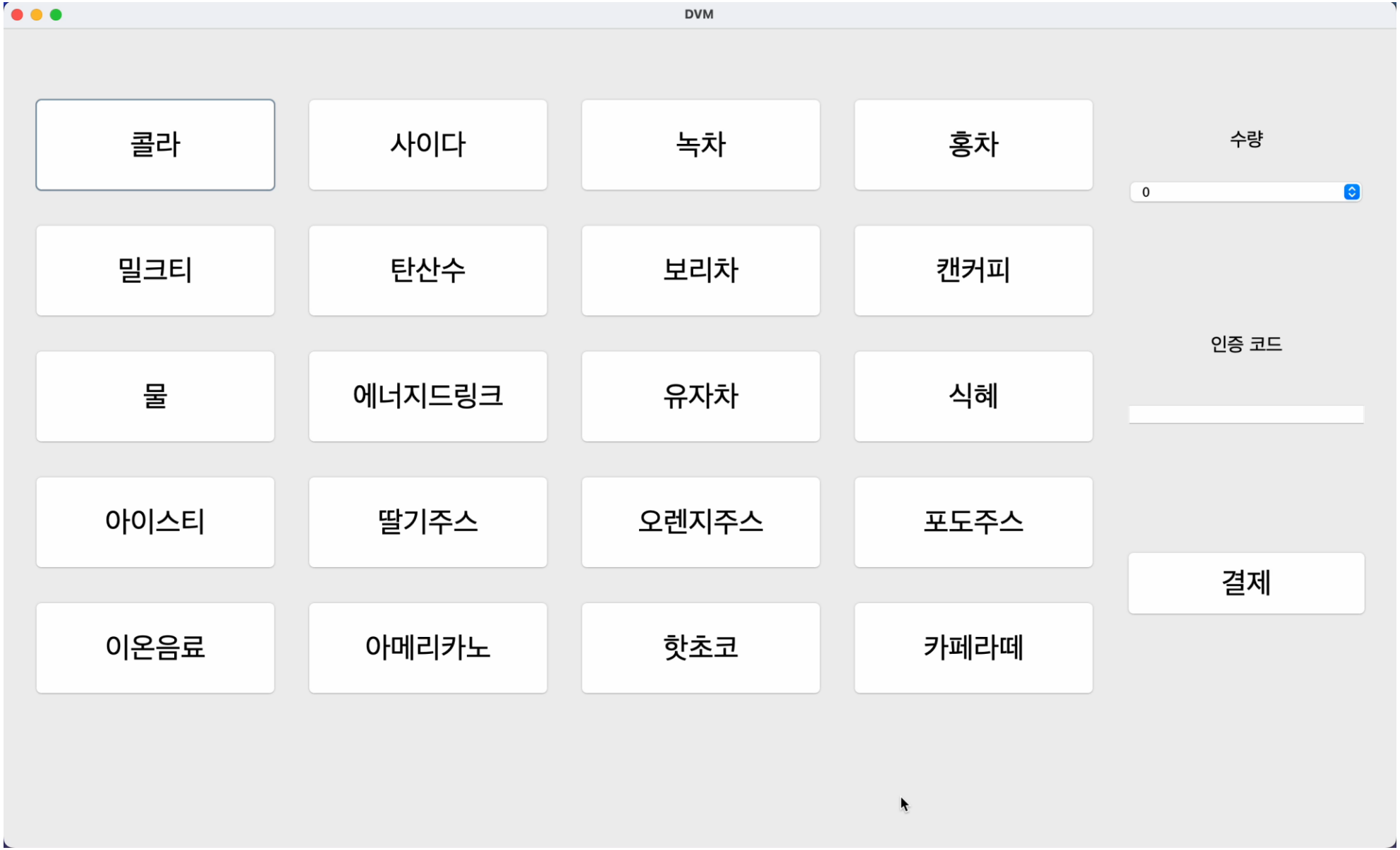
3. 시스템 동작 Demo 영상

잔액부족



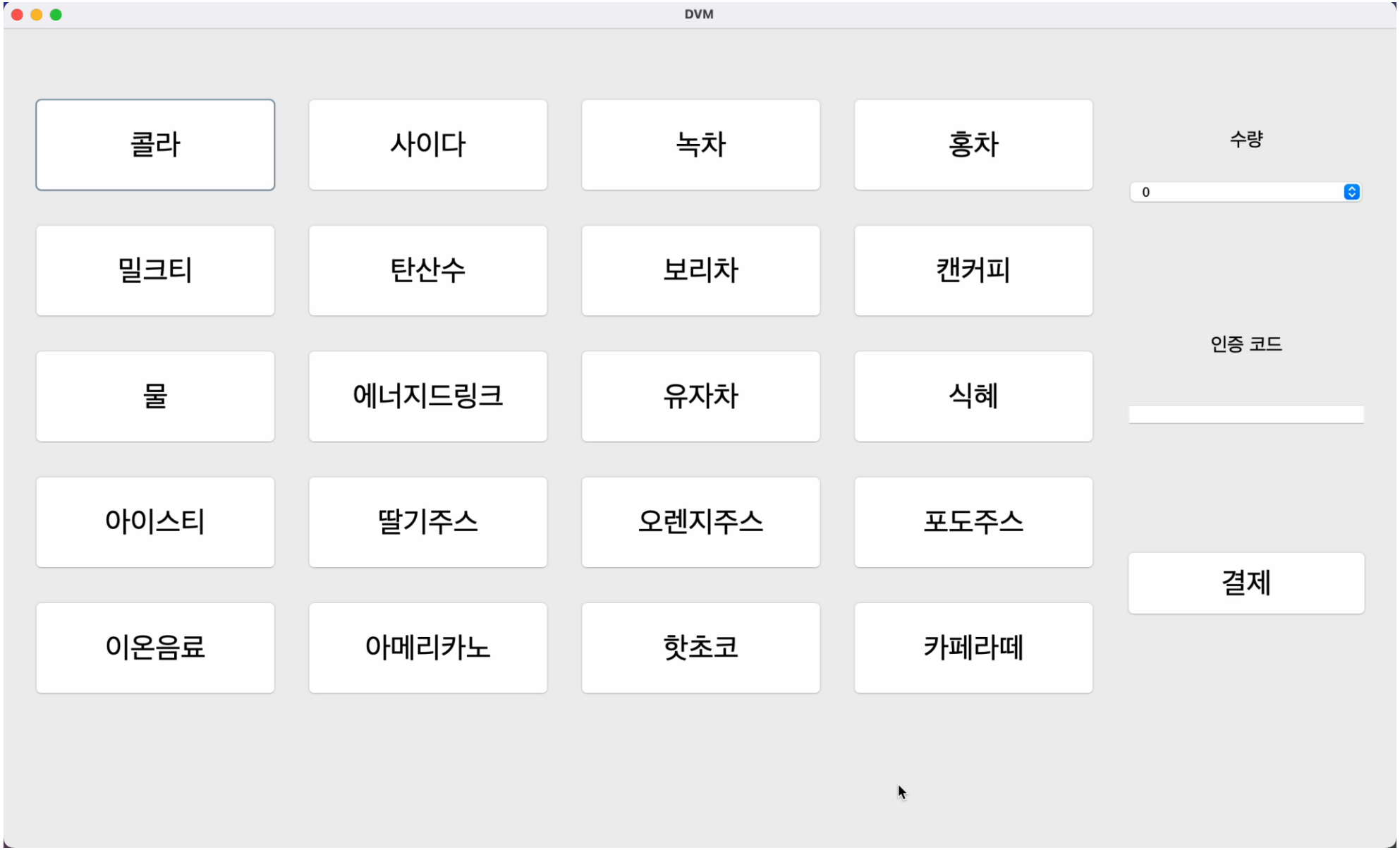
3. 시스템 동작 Demo 영상

재고부족



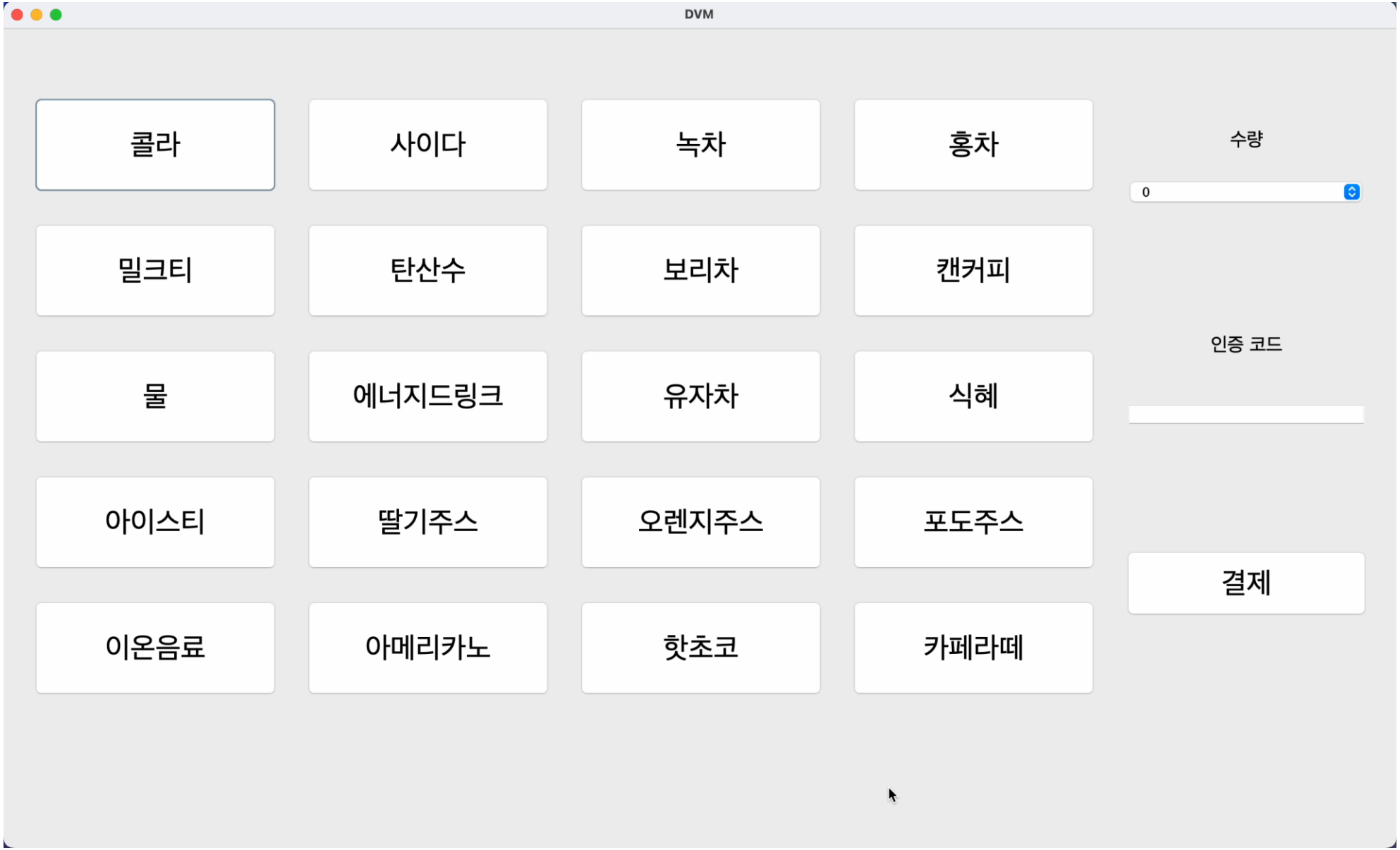
3. 시스템 동작 Demo 영상

선결제요청
및 성공



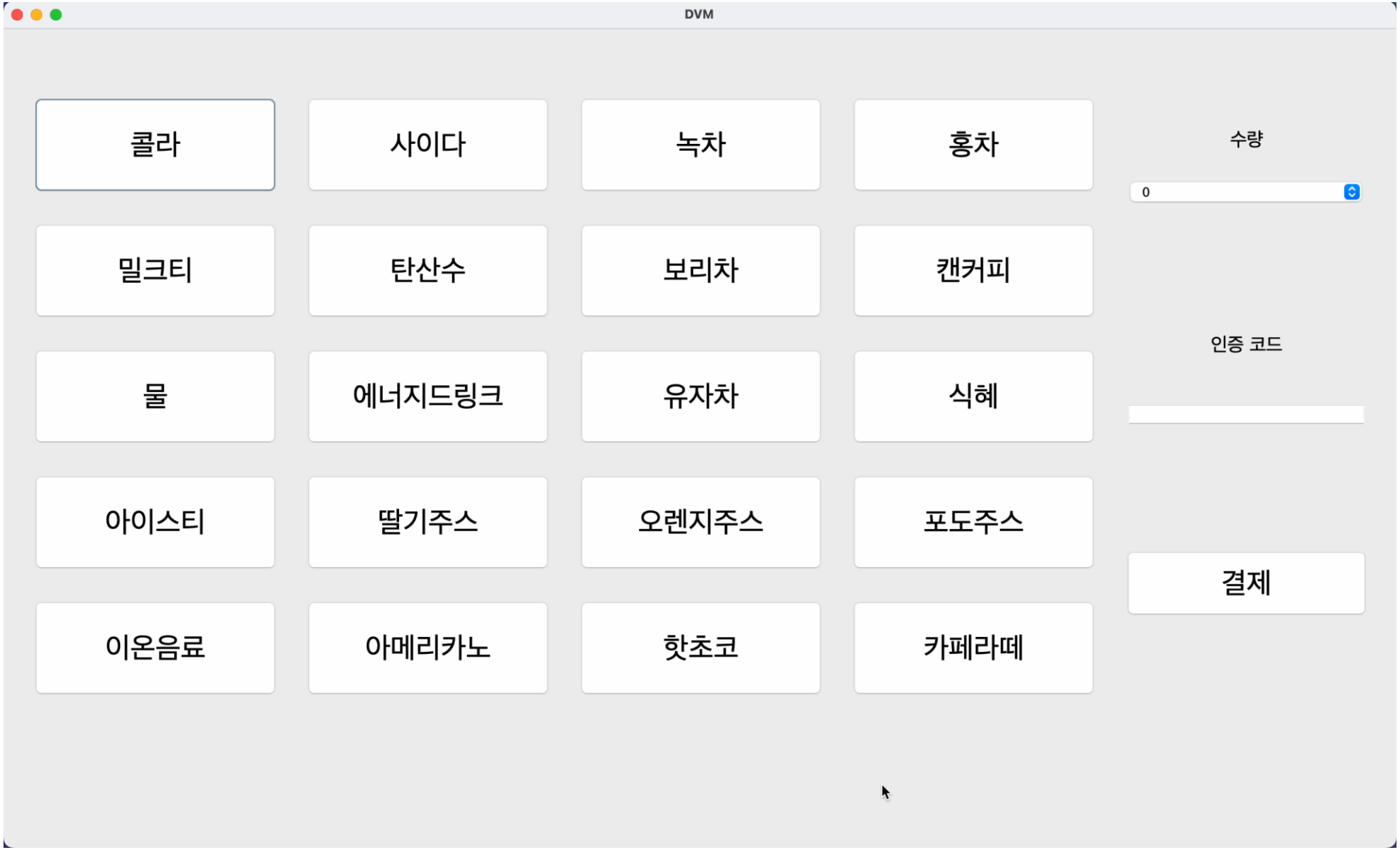
3. 시스템 동작 Demo 영상

선결제 거부



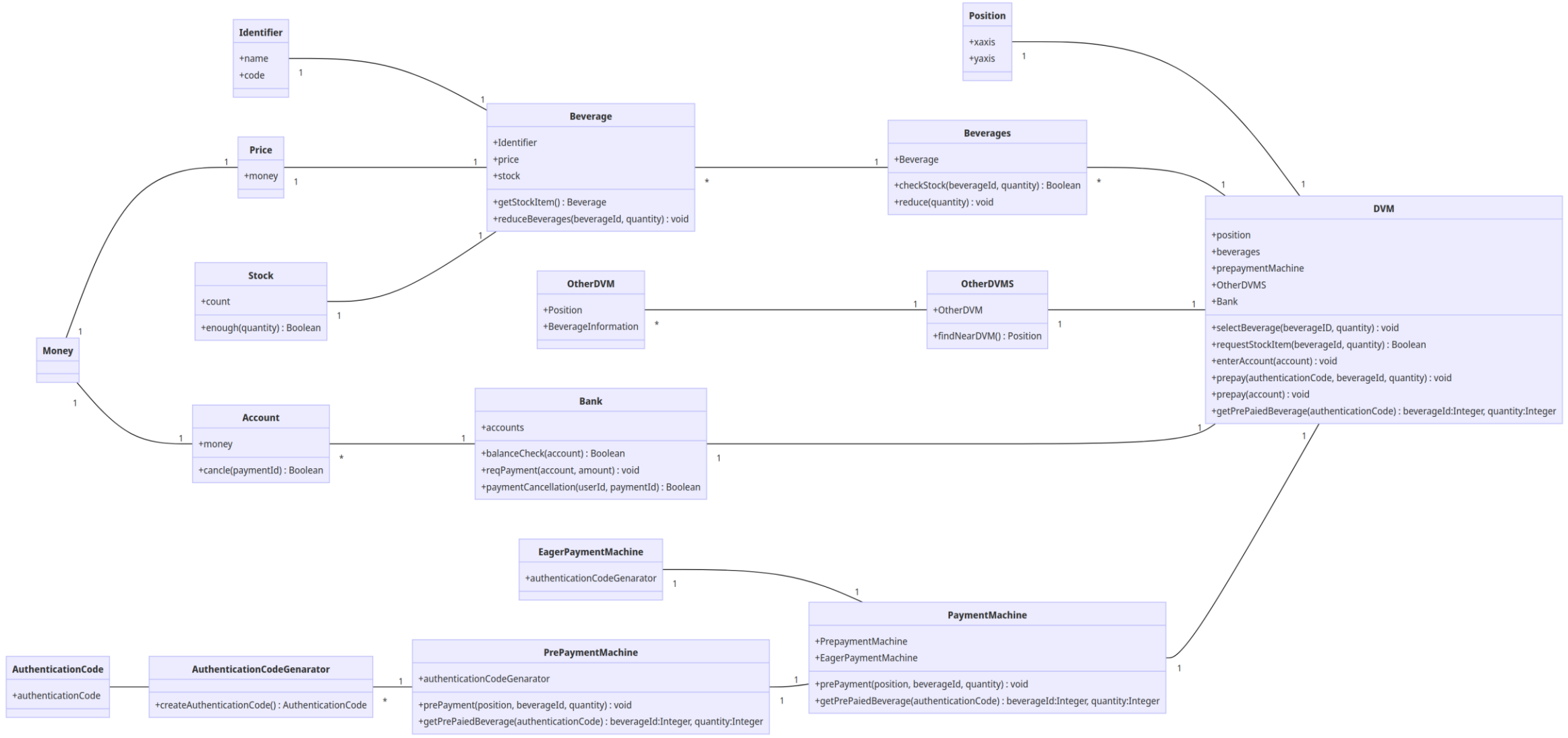
3. 시스템 동작 Demo 영상

다른 DVM에서
선결제한
음료 제공



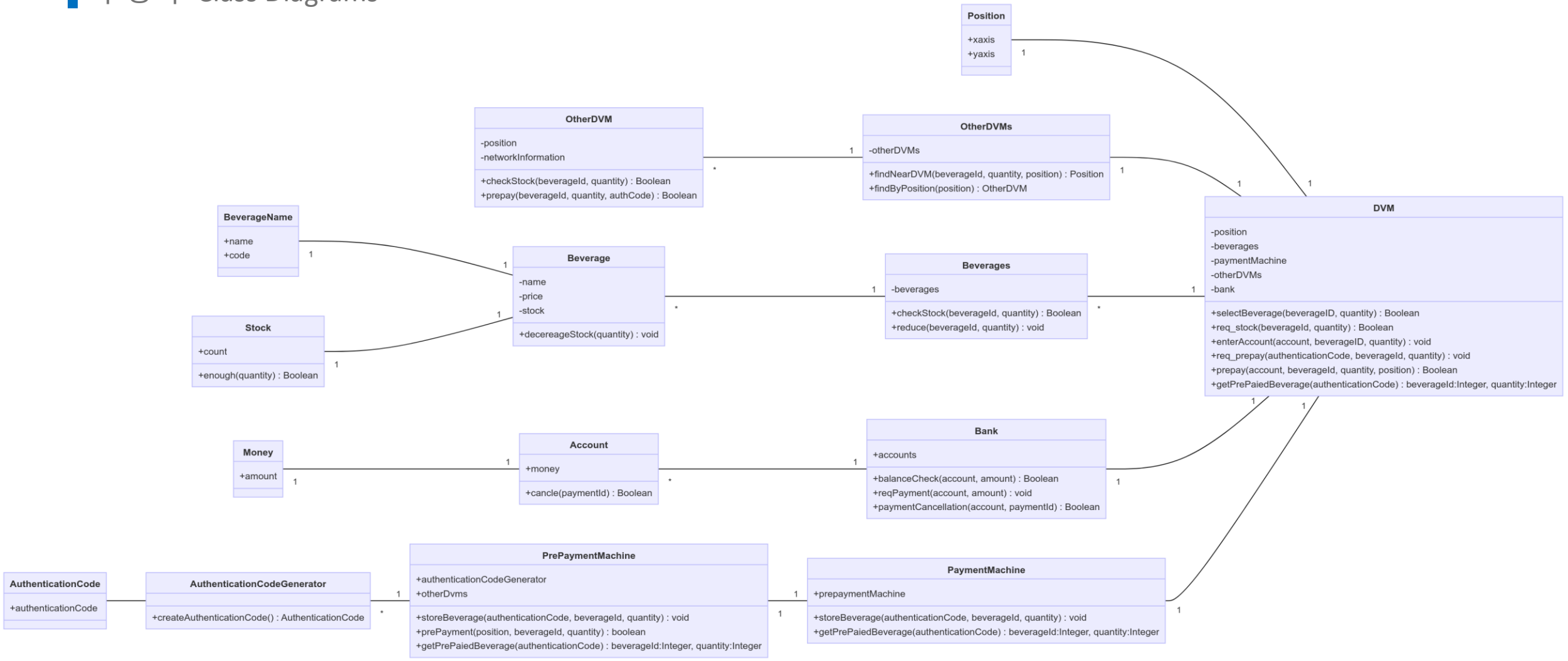
4. OOD (2040) 에서 변경/수정된 부분 정리

수정 전 Class Diagrams



4. OOD (2040) 에서 변경/수정된 부분 정리

수정 후 Class Diagrams



4. OOD (2040) 에서 변경/수정된 부분 정리

수정된 부분

클래스 구조 변경

- ‘Beverage’ 클래스가 기능적으로 ’BeverageName’과 ‘Stock’으로 세분화
- ‘BeverageName’은 음료의 식별 정보, ‘Stock’은 재고 관리를 담당

클래스 삭제

- ‘Identifier’, ‘Price’, ‘EagerPaymentMachine ' 클래스 삭제
- 이로 인해 관련된 다른 클래스들의 속성 및 메소드 변경

메소드 추가 및 변경

- OtherDVM 클래스에 checkStock, prepay 메소드가 추가
- OtherDVMs 클래스에 findByPosition 메소드 추가
- PaymentMachine, PrePaymentMachine 클래스에 storeBeverage 메소드 추가 및 변경

5. 구현 시 생각보다 어려웠던 점

서버에서 제공하는 API에 대한 예외처리 종류가 많이 다양하고 많아서 모두 처리하는 것에 어려움을 겪었고, 다른 DVM과 소통을하고 얻은 결과를 응답에 반영해주는 로직이 있기때문에 로컬에서 자체적으로 완벽하게 테스트를 해보는 것에 딜레이와 어려움이 있었습니다.

Socket 통신을 구현하는데 생각보다 매우 많은 시간이 소요되었습니다. Message Borker나 WebSocket 등, 평소 저수준의 기술을 직접 사용하기보다 오픈된 고수준의 라이브러리/프레임워크 위주로 사용하다보니 예상과 다른 부분이 많았습니다. 특히 다양한 예외 상황에 대한 분기 처리와 이로 인한 에러 관리가 힘들었습니다.

6. 구현 시 생각보다 쉬웠던 점

요구사항 자체가 복잡하지 않았던 만큼 전체적으로 간단할거라 생각했고, 도메인 부분은 역시 빠르게 마무리할 수 있었습니다.

클라이언트와 서버를 분리시켜 구현하였기때문에 서버와의 인터페이스만 알맞게 설정하였다면 클라이언트를 구현하는 것에 큰 어려움이 없었던 것 같습니다.

7. 객체지향개발방법론의 장단점 + 개인적인 소감들

201911193 우이산

OOAD 장점: 문서화를 명확히 한다는건, 구조를 제대로 알고 들어간다는 의미입니다. 구조를 제대로 안다는 건, 비로소 설계할 준비가 되었다고 생각합니다.

OOAD 단점: 직접 구현하기 전에는 보이지 않는 것들이 많습니다. 문서화를 마무리하고, 설계를 끝마치고, 구현을 시작한다는건 환상입니다.

느낀점: 갖가지 UML을 이렇게 본격적으로 사용해본게 처음이었습니다. 확실히 다이어그램을 그리며 구조를 점검하는 과정에서 편리함을 느꼈습니다. 특히 SSD가 정말 마음에 들었습니다. 다만, 설계를 마치고 구현을 시작한다는것이 환상이라는 사실도 느꼈기 때문에, 앞으로는 둘의 사이에서 균형을 잡으려 합니다.

7. 객체지향개발방법론의 장단점 + 개인적인 소감들

201911185 신동성

OOAD 장점: 확실히 코딩하기 전에 도메인에 대하여 잘 이해하고 들어갈 수 있었습니다. 또한 요구사항분석과 클래스 디자인 과정 등을 통해 무엇을 어떻게 구현하고 어떤 스킬을 쓸 지를 머릿속으로 안 상태로 코딩을 할 수 있다는 것이 가장 큰 장점인 것 같습니다.

OOAD 단점: 하지만 코딩을 하나도 안하고 설계를 하다 보니까 실제 코딩할 때 없어야 할 게 있었고 있어야 할 게 없었던 경우가 있었습니다.

느낀점: OOPT의 가장 큰 장점은 현실 세계의 문제에 대해서, 프로그래밍을 통한 해결 방법을 여러 단계로 나누어, 반복적인 단계를 거쳐, 점진적으로 구체화 시키고 발전시키는 것이라고 생각합니다. 하지만 이번 OOPT 팀프로젝트의 경우 모든 단계를 한 번씩만 진행하게 되어 이러한 장점을 살릴 수 없었던 것 같습니다. 회사가면 많이 하겠지만, 그래도 가기 전에 한번 쯤은 반복적인 개발을 할 기회가 있었으면 좋겠습니다

7. 객체지향개발방법론의 장단점 + 개인적인 소감들

202112340 이병직

OOAD 장점: 기능이 많고 거대한 프로젝트일수록, 그리고 협업을 통해 해당 프로덕트를 개발해야하는 경우에 요구사항 및 개발에 있어 애매한 것들을 확실하게 정립하고 개발을 시작하는 것이 효율적일 것이라고 생각합니다.

OOAD 단점: 구현을 하기 이전에 구현단계를 생각하며 설계하는 것은 비효율적일 수도 있다는 생각을 합니다.

느낀점: Static driven appraoch로 개발을 해왔기 때문에 해당 문제점을 명확히 알 수 있었고, 설계하는 방법을 배울 수 있어서 좋았던 것 같습니다. 앞으로의 프로젝트에서 OOAD의 일부를 활용할 수 있도록 하고자 합니다.

7. 객체지향개발방법론의 장단점 + 개인적인 소감들

201814263 김지우

솔직하게 OOD 부터의 과정은 너무 인적자원 낭비가 심한 과정이라 생각합니다. 실제 필드에서 코드를 구현하는 사람과 아키텍처가 분리돼 있는 점에서 오는 괴리가 OOD과정에서 크다고 생각합니다. 제가 좋다 생각하는 IT기업들은 아키텍처와 코더가 분리되어 있지 않습니다. 좋은 아키텍처를 생각하는 사람이 직접 구현도 하면서 부족한 부분을 알게 되고 지속적으로 개선해 나가는 방식이 매우 좋은 개발 방법이라 생각합니다. 에자일 소프트웨어 개발 선언의 다음 부분이 너무나 생각났습니다.

“ 포괄적인 문서보다 작동하는 소프트웨어를 ”

OOA단계의 Use Case까지만 만들고 바로 개발에 들어갔어야 한다고 저는 생각합니다. 실제로 저는 그런 방식으로 개발을 진행하기도 하고요, 개발에 시간적 여유가 더 있어야 TDD도 적용이 가능하고, UseCase기반의 ATDD도 가능하게 됩니다. OOD과정을 정말 해보고 싶다면, 먼저 개발을 어느 정도 몇 cycle 진행하여 프로덕트가 나와있는 상태에서, 문서 관리용으로 후에 작업하는 것이 적합하다고 생각합니다.

https://agilemanifesto.org/iso/ko/manifesto.html