DISTRIBUTED VENDING MACHINE (DVM)

2nd Cycle development

Project Team: Team 2

Date: 2021.05.28

Team Members :

201310513 황인우 / 201311255 최우석 / 201512265 박인우 / 201711306 박정현

Contents

Specification Review 개선 요구사항 반영 결과
Structural Coverage 개선 요구사항 반영 결과
Brute Force Testing 개선 요구사항 반영 결과

Specification Review

개선 요구사항 반영 결과

수정 요구사항

No	Stage	Page	Name	Review
1	1000	3	Functional-Requirements add 1	DVM_PFR Page 2 의 1.2 Requirements 의 1 번째 항목에서 '총 음료의 개수는 20 종류이다.'에 대한 언급이 없으므로 추가 필요
2	1000	3	Functional-Requirements add 2	DVM_PFR Page 2 의 1.2 Requirements 의 2 번째 항목에서 '한 자판기는 7 종류의 음료를 판매한다.'에 대한 언급이 없으므로 추가 필요
3	1000	3	Functional-Requirements Refine 1	5 번째 항목에서 다른 위치의 자판기 정보 제공하는 조건에 다른 자판기에는 해당 상품에 대한 재고가 있다는 조건 추가 필요
4	1000	3	Non-Functional Requirements add 1	DVM_PFR Page 3 의 1.5 Assumptions and dependencies 의 1 번째 항목에서 '자판기의 개수는 최대 10 개로 가정한다.'에 대한 언급이 없으므로 추가 필요
5	1000	3	Non-Functional-Requirements Refine 1	'자판기의 조작이 쉬워야한다', '재고의 관리가 용이해야 한다.', '자판기가 튼튼해야 한다.'와 같은 모호한 표현 수정 필요

요구사항 반영 후 문서 내용

: Project Planning Activity 1001

Functional Requirements

- 시스템이 제공할 총 음료의 개수는 20종류이다.
- 한 자판기에서는 7종류의 음료를 판매한다.
- 시스템은 상품의 이름 및 가격, 물량 수을 제공한다.
- 시스템은 사용자의 결제 수단에 따라 상품 금액을 계산해 처리한다.
- 시스템은 사용자가 상품을 고를 수 있도록 한다.
- 시스템은 사용자가 고른 상품을 재고에서 처리하고 사용자에게 제공한다.
- 시스템은 사용자가 고른 상품이 현 자판기 재고에 없을 경우 해당 상품의 재고를 보유한 다른 위치의 자판기 정보를 제공한다.
- 시스템은 다른 자판기의 제품을 선결제할 수 있는 서비스를 제공한다.

Non-Functional Requirements

- 자판기의 개수는 최대 10개로 가정한다.
- ~~자판기 조작이 쉬워야 한다.¶~~
~~자판기의 관리가 쉬워야 한다.¶~~
~~재고의 관리가 용이해야 한다.¶~~
~~자판기가 튼튼해야 한다.~~

수정 요구사항

6	1000	5	Functional Requirement – Description Refine	Pay by Cash, Pay by Card 의 '가격합'이라는 표현과 Calculate Price(Cash), Calculate Price(Card)의 '상품들의'라는 표현을 보면 다중 선택이 가능하다는 의미가 될 수 있으므로 수정 필요

요구사항 반영 후 문서 내용

: Project Planning Activity 1003

Pay by Cash	사용자가 상품의 가격합을 현금으로 결제한다.
Pay by Card	사용자가 상품의 가격합을 카드로 결제한다.
Calculate Price (Cash)	시스템이 선택된 상품들의 가격을 계산하고 거스름돈을 산출한다.
Calculate Price (Card)	시스템이 선택된 상품들의 가격을 계산하고 사용자의 카드를 기반으로 결제를 시도한다.

수정 요구사항

7	1000	5	Functional Requirement - Login	cf. switch 가 무슨 뜻인지 이해할 수 없으므로 삭제 또는 추가 설명 필요

요구사항 반영 후 문서 내용

: Project Planning Activity 1003

Login	관리자가 로그인을 시도한다. ~~cf. Switch~~

수정 요구사항

8	1000	6	R1.1 name	Page 5 의 Function 1 번째 항목은 Show Item 인데, Show Menu 로 나와있으므로 수정 필요
9	1000	8	R1.5 Category	Page 6 의 R1.5 Category 는 Evident 인데, Hidden 으로 나와있으므로 수정 필요

요구사항 반영 후 문서 내용

: Project Planning Activity 1003

Show Menu~~Item~~	시스템이 현재 자판기에서 판매중인 제품 및 재고 정보를 보여준다.	
R1.5	Choose item or code	Hidden~~Evident~~

수정 요구사항

10	1000	11	Use Case 15 Description	Page 14 의 Test Case 15 에는 '거리순서대로'라는 조건에 맞는 경우 Test Pass 라고 나와있는데, 해당 조건에 대한 설명이 없으므로 추가 필요

요구사항 반영 후 문서 내용

: Project Planning Business Use Case 15. Give Item Location

Use Case	15. Give Item Location
Actors	System
Description	- 시스템이 사용자가 원하는 제품이 존재하는 DVM들의 위치정보를 현재 DVM으로부터 거리가 가까운 순서대로 알려줍니다.

Project Planning Activity 1009

15. Give Item Location	재고가 있는 DVM의 위치정보를 제대로 알려주는지 확인한다. 위치정보를 현재 DVM으로부터 거리가 가까운 순서대로 알려주는지 확인한다.

수정 요구사항

11	SRS		Page Number	Page Number 추가 필요

요구사항 반영 후 문서 내용

: SRS 문서에 Page Number를 추가하였습니다.

수정 요구사항

12	2030	13	Term Name - card	Stage1000 의 Page 7 의 Credit Description 과 동일하지만, Card 라고 나와있으므로 수정 필요

요구사항 반영 후 문서 내용

: Project Planning Activity 1004

Card~~Credit~~	카드 결제 시 사용되는 카드

OOA Activity 2034

Card	카드 결제 시 사용되는 카드

수정 요구사항

13	2030 SRS	3 5	Use Case 1 – Overview 1	세 가지 주요 기능이 무엇인지 정확히 명시하도록 수정 필요

요구사항 반영 후 문서 내용

: OOA, SRS Use Case 1. Show Menu

Overview	사용자에게 상품 구매, 쿠폰 사용, 관리자 로그인 기능~~세가지 주요 기능~~ 중 하나를 선택할 수 있는 초기 선택 메뉴를 보여줍니다.
Typical Courses of Events	(U): User, (S): System 1. (S) System이 사용자가 상품 구매, 쿠폰 사용, 관리자 로그인 기능 중 하나를 선택할 수 있는 화면을~~제품 선택 활성화하기, 쿠폰 사용, Login을 선택할 수 있도록~~ 출력합니다.

수정 요구사항

14	2030 SRS	3 5	Use Case 1 – Overview 2	Stage1000 의 Page 9dml Description 에는 '시스템이 상시 DVM 에 있는 모든 제품의 이름과 재고 현황을 알려줍니다.'라고 나와있지만, '~ 초기 선택 메뉴를 보여줍니다.'로 불일치하는 설명 수정 필요

요구사항 반영 후 문서 내용

: Project Planning Business Use Case 1. Show Menu

Use Case	1. Show Menu
Actors	System
Description	- 시스템이 사용자에게 최초로 기능 선택 메뉴를 보여줍니다.~~상시 DVM에 있는 모든 제품의 이름과 재고 현황을 알려줍니다~~

수정 요구사항

15	2030 SRS	3 5	Use Case 2	같은 의미의 용어로 보여지는 '제품', '상품'이 혼용되어 사용되고 있으므로 통일하는 방향으로 수정 필요

요구사항 반영 후 문서 내용

: Project Planning, OOA, SRS, OOD, SDS 문서에서 사용되는 '제품', '상품' 단어들을 모두 '상품'으로 통일했습니다.

수정 요구사항

16	2030 SRS	9 11	Use Case 13 Type	Typical Courses of Events 에 System 의 Event 만 존재하므로 Hidden 으로 수정 필요
17	2030 SRS	9 11	Use Case 14 Type	Typical Courses of Events 에 System 의 Event(Another DVM)만 존재하므로 Hidden 으로 수정 필요

요구사항 반영 후 문서 내용

: OOA, SRS Use Case 13. Get Another DVM Info

Use Case	13. Get Another DVM Info
Actor	System
Purpose	시스템이 다른 시스템에 정보(재고현황, 위치)를 요청합니다.
Overview	사용자가 요청한 재고가 없는 상품에 대한 정보를 얻기위해 다른 시스템에 정보(재고현황, 위치)를 요청합니다.
Type	Hidden~~Evident~~
Cross Reference	12. Get None Item Location

OOA, SRS Use Case 14. Give DVM Info

Use Case	14. Give DVM Info
Actor	Another DVM
Purpose	타 DVM이 현 DVM에게 요청한 재고의 유무를 알려줍니다.
Overview	현 DVM에게 정보(해당 상품 재고, 타 DVM 위치)를 반환합니다.
Type	Evident
Cross Reference	13. Get Another DVM Info 15. Give Item Location
Pre-Requisites	타 DVM에게 Get Another DVM Info로 요청을 받아야 합니다
Typical Courses of Events	(S): System, (D): Another DVM 1. (D) System에게 재고와 위치를 요청합니다 2~~4~~. (S~~D~~) 본 기기의 해당 재고 유무를 파악합니다. 3~~2~~. (S~~D~~) 기기의 재고와 위치를 Another DVM~~System~~에게 반환합니다.

수정 요구사항

18	2030 SRS	11 12	Use Case Number	Stage1000 의 Use Case 는 19 개로 17. Manage Kind of Product 가 누락된 것이라면 수정 필요 (누락이 아닌 삭제의 경우, Stage1000 수정 필요)

요구사항 반영 후 문서 내용

: Project Planning Activity 1006

~~R4.1~~¶	~~Manage Kind of Product~~¶	~~Evi~~
R4.~~2~~	Manage Amount	Evi
R4.~~3~~	Manage Amount Cash	Evi

~~Use Case~~¶	~~17. Manage Kind of Product~~¶
~~Actors~~¶	~~Administrator~~¶
~~Description~~¶	~~- 관리자가 시스템에 새로운 제품을 등록하거나 제거합니다.~~¶

Use Case	~~1~~7~~8~~. Manage Amount
Actors	Administrator
Description	- 관리자가 현재 DVM에 등록된 제품의 재고를 추가 혹은

수정 요구사항

19	2030	13	Refine Glossary	인증 코드라는 말을 주로 사용하지만 Glossary 에는 코드에 대해서만 정의되어 있으므로 수정 또는 통일 필요

요구사항 반영 후 문서 내용

: Project Planning, OOA, SRS, OOD, SDS 문서에서 사용되는 '코드', '인증 코드' 단어들을 모두 '코드'로 통일했습니다. (GUI의 '쿠폰 사용' 단어의 경우에는 수정하지 않고 유지)

(U): User, (S): System
1. (U) User가 쿠폰 사용 버튼을 누릅니다.
2. (S) System이 ~~인증~~코드를 입력받을 창을 Software-GUI 화면에 출력합니다.
3. (U) User가 ~~인증~~ 코드를 입력합니다
4. (S) System이 ~~인증~~코드를 받으면 8. Code Validation으로 넘어갑니다.

(S): System
1. (S) ~~인증~~ 코드의 유효성을 모든 DVM에서 체크합니다.
2. (S) 유효한 코드일 경우 ~~인증~~ 코드로부터 확인된 상품의 재고가 해당 시스템에 존재하는지 확인합니다.
3. (S) 상품의 재고가 있을 경우, 해당 코드를 사용한 것으로 처리하고 해당 정보를 다른 시스템에게도 알립니다.
4. (S) Give Item으로 넘어갑니다

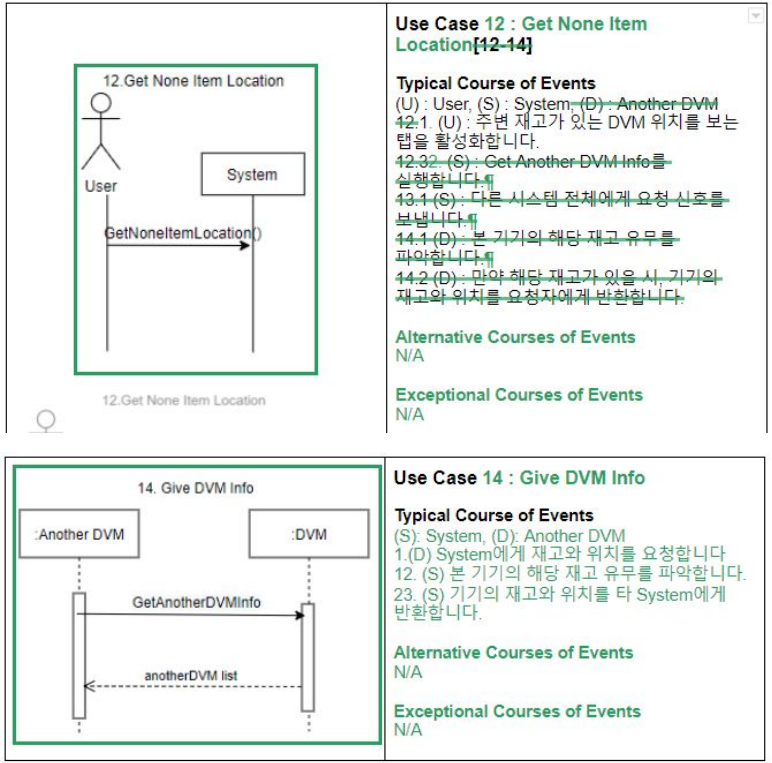
A2.1: 유효하지 않은 코드일 경우, 유효하지 않은 코드라는 메시지를 Software-GUI 화면에 출력하고 Use Code로 돌아가 다시 ~~인증~~ 코드를 입력하도록 합니다.

수정 요구사항

20	2030	16	System Sequence Diagram	Use Case [12-14]로 하나의 Sequence Diagram 만 표현되어있는데, Use Case 별로 분리하여 작성 필요

요구사항 반영 후 문서 내용

: OOA Activity 2035

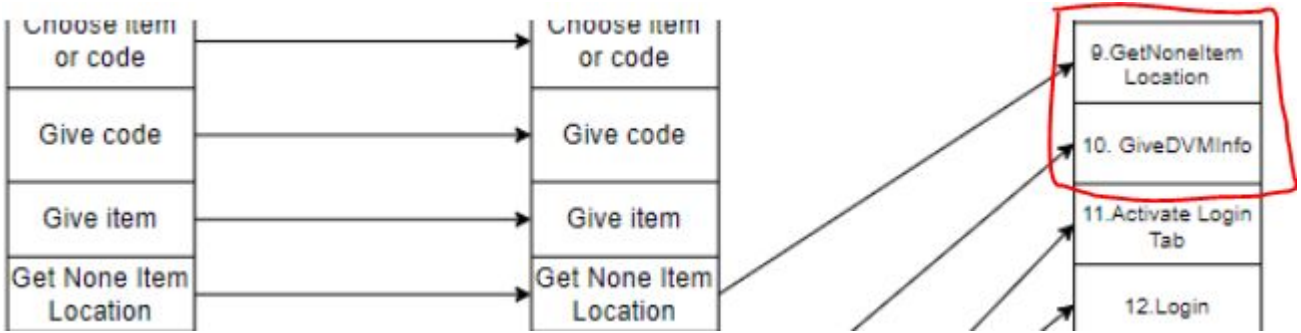


수정 요구사항

21	2030	21	Traceability System Operation	9. GetNoneItemLocation( ), 10. GiveDVMInfo( )에 대해서만 뒤에 '( )'표시를 붙여 함수형로 표현하였는데, 다른 항목에 대해도 동일하게 함수형으로 처리하거나 '( )'표시 삭제 필요

요구사항 반영 후 문서 내용

: OOA Activity 2039



수정 요구사항

22	SRS	14	Other Requirement	1 번째 항목의 '자판기 내부의' 다음에 이어지는 문장이 없으므로 삭제 또는 수정 필요

요구사항 반영 후 문서 내용

: SRS 3.6 Other requirements

3.6 Other requirements¶

~~자판기 내부의~~

- 자판기가 보유한 제품의 상태 (유통기한, 파손 등) 여부와 사용자의 제품 수령 후 제품 상태에 따른 후속 처리와 같은 판매 이외의 과정에 대해서는 고려하지 않는다.
- DVM 내부 모든 음료의 상태는 판매 가능한 상태이다

수정 요구사항

23	2030	18-20	System Test Case	거의 모든 항목에서 '정상적으로', '제대로'라는 표현을 사용하는데 어떤 상황이 정상적인지 파악 불가하므로 명확한 용어 사용 필요

요구사항 반영 후 문서 내용

: Stage 2030 OOA Activity 2038 Refine System Test Case에서 '정상적으로', '제대로'와 같은 모호한 표현들을 삭제하였습니다.

사용자가 상품~~제품~~ 선택을 활성화시킬 경우 자판기에 등록되어 있는 상품 목록이 출력되는지~~활성화가 정상적으로 진행되는지~~ 확인한다.
시스템이 등록된 상품~~제품~~들의 이름과 재고 현황을 ~~정상적으로~~ 출력하는지 확인한다.
재고가 있는 상품을 선택했을 때 ~~정상적으로~~ 결제 선택 화면으로 넘어가는지 확인한다.
재고가 없는 상품을 선택했을 때 사용자에게 선결제 확인 메시지를 출력하고 확인 버튼을 누르면 재고가 없는 상품에 대한 기능 선택 화면으로 넘어가는지 확인한다.

~~사용자가 재고가 없는 상품을 고르는 탭을 활성화시킬 경우 정상적으로 활성화되는지 확인한다.~~
사용자가 재고가 없는 상품을 보유한 다른 DVM의 위치 보기를 재고가 없는 상품에 대해서만 수행~~선택~~ 가능한지 확인한다.

사용자가 투입한 카드가 시스템에 ~~정상적으로~~ 투입되는지 확인한다.

수정 요구사항

24	2040	8	Use Case 10 – Cross Reference	Stage2030 Use Case 10 의 내용과 불일치하므로 삭제 또는 수정 필요
25	2040	9	Use Case 11 – Cross Reference	Stage2030 Use Case 11 의 내용과 불일치하므로 삭제 또는 수정 필요

요구사항 반영 후 문서 내용

: OOA Use Case 10. Give Code, 11. Give Item

Use Case	10.Give Code
Actor	System
Purpose	사용자가 선택한 상품에 대해 ~~인증~~ 코드를 제공합니다.
Overview	사용자가 선택한 상품을 구입할 수 있는 ~~인증~~ 코드를 제공합니다.
Type	Hidden
Cross Reference	2. Choose item 6. Calculate Price(Cash) 7. Calculate Price(Card) 9. Choose item or code

Use Case	11.Give Item
Actor	System
Purpose	사용자가 선택한 상품을 제공합니다.
Overview	사용자가 선택한 상품을 제공하고, 제공한 수량 만큼을 시스템 내부 재고량 에서 뺍니다.
Type	Hidden
Cross Reference	2. Choose item 6. Calculate Price(Cash) 8. Code validation 9. Choose item or code

수정 요구사항

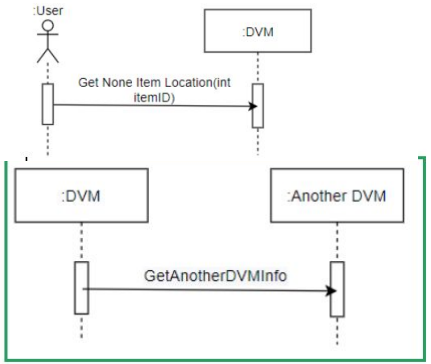
26	2040 SDS	15-17 6-8	Interaction Diagram	Use Case 별로 Interaction Diagram 분리 작성 필요

요구사항 반영 후 문서 내용

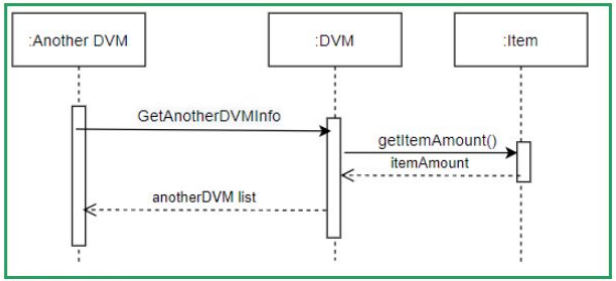
: OOD Activity 2043

12~~-15~~. Get None Item Location~~ & Get Another DVM Info & Give DVM Info & Give Item Location~~¶

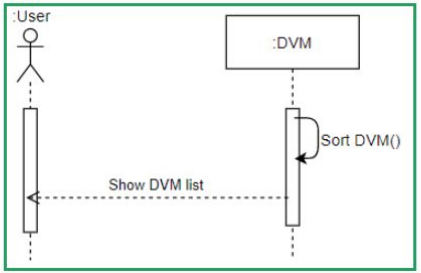
13.



14. Give DVM Info



15. Give Item Location



수정 요구사항

27	2040	16	Use Case 16 - Event	Typical Courses of Events 에 1 번 Event 뒤에 아이디와 비밀번호 출력에 대한 System 의 Event 가 누락되어 있으므로 추가 필요

요구사항 반영 후 문서 내용

: OOD Use Case 16. Login

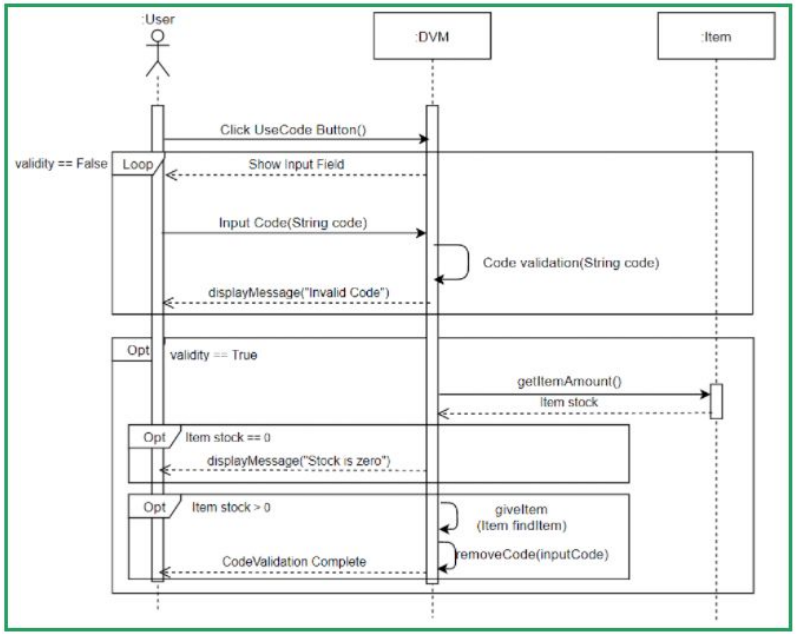
Use Case	16. Login
Actor	Administrator
Purpose	시스템 관리자가 관리자임을 확인해 관련 작업을 수행할 수 있도록 합니다.
Overview	시스템 관리자가 아이디와 비밀번호를 입력하고 이를 확인하여 관리자 권한을 부여합니다.
Type	Evident
Cross Reference	17. Manage Amount 18. Manage Amount Cash
Pre-Requisites	N/A
Typical Courses of Events	(A): Administrator, (S) : System 1. (A) 관리자가 Software-GUI 화면의 Login 버튼을 누릅니다. 2. (S) 시스템이 로그인 Software-GUI를 띄웁니다 3. (A) 관리자가 Software-GUI 화면의 아이디와 비밀번호를 입력합니다. 4. (S) 시스템이 등록된 아이디와 비밀번호와 비교하고 일치할 경우 관리자 메뉴(제품재고 관리, 현금 관리)를 Software-GUI 화면에

수정 요구사항

28	2040 SDS	16 7	Interaction Diagram – Use Code & Code Validation	CodeTable 의 경우 Class Diagram 에서는 객체가 아닌데, Interaction Diagram 에서는 객체인 것처럼 작성되어 있음

요구사항 반영 후 문서 내용

: OOD Activity 2043



수정 요구사항

29	2040	21	Traceability line	연결하는 화살표가 많이 겹쳐서 내용 파악이 불가하므로 넘버링을 통해 표로 작성 요청

요구사항 반영 후 문서 내용

: Traceability Analysis 문서를 Table로 추가 작성했습니다.

Traceability Analysis (Table)

System Tests	Matching Functional Requirements
1-1 최초 화면 GUI 출력 Test	Show Menu
2-1 상품 리스트 GUI 출력 Test	Choose Item
2-2 상품 리스트 GUI 출력 Test	
2-3 결제방법 선택 GUI 출력 Test	
2-4 상품 재고가 없는 경우 안내 GUI 출력 Test	
3-1 현금 결제 GUI 출력 및 기능 수행 Test	Pay By Cash
3-2 현금 결제 GUI 출력 및 기능 수행 Test	
4-1 카드 결제 GUI 출력 및 기능 수행 Test	Pay By Card
4-2 카드 결제 GUI 출력 및 기능 수행 Test	
5-1 쿠폰 사용 GUI 출력 및 기능 수행 Test	Use Code
5-2 쿠폰 사용 GUI 출력 및 기능 수행 Test	

System Tests	Matching Functional Requirements
6-1 현금 결제 기능 수행 Test	Calculate Price(Cash)
6-2 현금 결제 기능 수행 Test	
6-3 현금 결제 기능 수행 Test	
7-1 카드 결제 기능 수행 Test	Calculate Price(Card)
7-2 카드 결제 기능 수행 Test	
7-3 카드 결제 기능 수행 Test	
8-1 쿠폰 결제 기능 수행 Test	Code validation
8-2 쿠폰 결제 기능 수행 Test	
8-3 쿠폰 결제 기능 수행 Test	
9-1 현금, 카드 결제 및 쿠폰 사용 기능 수행 Test	Choose Item or code
9-2 현금, 카드 결제 및 쿠폰 사용 기능 수행 Test	
9-3 현금, 카드 결제 및 쿠폰 사용 기능 수행 Test	
10-1 상품 재고가 없는 경우의 결제 기능 수행 Test	Give code

System Tests	Matching Functional Requirements
11-1 상품 재고가 없는 경우의 결제 기능 수행 Test	Give item
11-2 상품 재고가 없는 경우의 결제 기능 수행 Test	
12-1 상품 재고가 없는 경우 안내 GUI 출력 Test	Get None Item Location
12-2 상품 재고가 없는 경우 안내 GUI 출력 Test	
13-1 현재 DVM에 재고 없는 물품에 대한 다른 DVM 재고 안내 기능 수행 Test	Get Another DVM Info
14-1 현재 DVM에 재고 없는 물품에 대한 다른 DVM 재고 안내 기능 수행 Test	Give DVM Info
14-2 현재 DVM에 재고 없는 물품에 대한 다른 DVM 재고 안내 기능 수행 Test	
14-3 현재 DVM에 재고 없는 물품에 대한 다른 DVM 재고 안내 기능 수행 Test	
15-1 현재 DVM에 재고 없는 물품에 대한 다른 DVM 재고 안내 기능 수행 Test	Give Item Location
15-2 현재 DVM에 재고 없는 물품에 대한 다른 DVM 재고 안내 기능 수행 Test	
15-3 현재 DVM에 재고 없는 물품에 대한 다른 DVM 재고 안내 기능 수행 Test	
15-4 현재 DVM에 재고 없는 물품에 대한 다른 DVM 재고 안내 GUI 출력 Test	

System Tests	Matching Functional Requirements
16-1 관리자 로그인 GUI 출력 Test	Login
16-2 관리자 로그인 기능 수행 Test	
16-3 관리자 로그인 기능 수행 Test	
17-1 DVM 물품재고 관리 GUI 출력 Test	Manage Amount
17-2 DVM 물품재고 관리 기능 수행 Test	
18-1 DVM 현금 관리 GUI 출력 Test	Manage Amount Cash
18-2 DVM 현금 관리 기능 수행 Test	

Functional Requirements	Matching Use Cases
Show Menu	Show Menu
Choose Item	Choose Item
Pay by Cash	Pay by Cash
Pay by Card	Pay by Card
Use code	Use code
Calculate Price(Cash)	Calculate Price(Cash)
Calculate Price(Card)	Calculate Price(Card)
Code Validation	Code Validation
Choose Item or code	Choose Item or code

Functional Requirements	Matching Use Cases
Give code	Give code
Give Item	Give Item
Get None Item Location	Get None Item Location
Get Another DVM Info	Get Another DVM Info
Give DVM Info	Give DVM Info
Give Item Location	Give Item Location
Login	Login
Manage Amount	Manage Amount
Manage Amount Cash	Manage Amount Cash

Use Cases	Matching Operations in System Sequence diagrams
Show Menu	-
Choose Item	1. Activate Choices
	2. Choose Item
Pay by Cash	3. Pay By Cash
	4. Input Cash
Pay by Card	5. Pay By Credit
	6. Insert Card
Use code	7. Use code
	8. Input Code
Calculate Price(Cash)	3. Pay By Cash
	4. Input Cash
Calculate Price(Card)	5. Pay By Credit
	6. Insert Card

Use Cases	Matching Operations in System Sequence diagrams
code validation	8. Input Code
Choose Item or code	2. Choose Item
Give code	-
Give Item	-
Get None Item Location	9. Get None Item Location
Get Another DVM Info	
Give DVM Info	10. Give DVM Info
Give Item Location	9. Get None Item Location

Use Cases	Matching Operations in System Sequence diagrams
Login	11. Activate Login Tab
	12. Login
Manage Amount	13. Activate Manage Amount
	14. Manage Amount
Manage Amount Cash	15. Activate Manage Cash
	16. Manage Cash

Operations in System Sequence Diagrams	Matching Operations in Interaction Diagrams
1. Activate Choices	Click choose item button()
2. Choose Item	Choose Item()
	getItemAmount()
3. Pay By Cash	Click payByCash button()
4. Input Cash	InputCash()
	Calculate Price Cash()
	getItemPrice()
	reduceAmount()
5. Pay By Credit	Click PayByCard button()
6. Insert Card	getItemPrice()
	Insert Card()
	Calculate Price Card()
	reduce Amount()

Operations in System Sequence Diagrams	Matching Operations in Interaction Diagrams
7. Use code	Click UseCode button()
8. Input Code	Input code()
	Code validation()
	reduce Amount()
9. GetNoneItemLocation	getNoneItemLocation()
10. GiveDVMInfo	get AnotherDVMInfo()
11. Activate Login Tab	Click LoginTab button()
12. Login	Login()
13. Activate Manage Amount	Click ManageAmount Button()
14. Manage Amount	Manage Amount()
	Change Amount()
15. Activate Manage Cash	Click Manage Amount Cash button()
16. Manage Cash	Manage Amount Cash()

Operations in Interaction Diagrams	Matching Methods	Matching Classes
Click Choose Item Button()	getItemAnount()	Item
Choose Item()	getItemList()	DVM
	getItemFromName(String)	
	giveItem(Item)	
	giveCode(Item)	
	getItemName()	Item
getItemAnount()	getItemAmount()	
Click payByCash Button()	inputCash(String, Item)	DVM
Input Cash()		
Calculate Price Cash()	CheckTotalCash(int)	
	calculatePriceCash(int, Item)	Payment
	isSufficient(int, Item)	
getItemPrice()	getItemPrice()	Item

Operations in Interaction Diagrams	Matching Methods	Matching Classes
Click payByCard Button()	insertCard(String, Item)	DVM
Insert Card()		
Calculate Price Card()	isSufficient(int, Item)	Payment
	calculatePriceCard(String,Item)	
	consumeCard(String, int)	
	isValidCard(String)	
Click UseCode Button()	inputCode(String)	DVM
Input Code()		
Code Validation()	removeCode(String)	
	codeValidation(String)	
	getItemFromCode(String)	
reduceAmount()	reduceAmount()	Item

Operations in Interaction Diagrams	Matching Methods	Matching Classes
getNoneItemLocation()	getDist(void)	DVM
	setDist(double)	
	getRegion()	
	getLocation()	
	getNoneItemLocation(String)	
getAnotherDVMInfo()	getAnotherDVMInfo(String)	
Click Login Tab Button()	login(String, String)	
Login()		
Click Manage Amount Button()	manageAmount(String, int)	
ManageAmount()		
ChangeAmount()	changeAmount	Item
Click Manage Amount Cash Button()	manageAmountCash(int)	DVM
Manage Amount Cash()		

Classes	Matching Unit Tests
DVM	getItemFromName Test
	getAnotherDVMInfo Tests
	removeCode Tests
	SortDVM Test
	getNoneItemLocation Test
	inputCode Tests
	CodeValidation Tests
	getItemFromCode Tests

Classes	Matching Unit Tests
DVM	login Tests
	manageAmount Tests
	manageAmountCash Tests
	inputCash Tests
	checkTotalCash Tests
	insertCard Tests
	giveItem Tests
	giveCode Tests

Classes	Matching Unit Tests
Payment	calculatePriceCash Tests
	isSufficient Tests
	calculateChange Test
	calculatePriceCard Tests
	consumeCard Test
	isValidCard Tests
Item	changeAmount Tests
	reduceAmount Tests

수정 요구사항

30	SDS		Page Number	Page Number 추가 필요

요구사항 반영 후 문서 내용

: SDS 문서에 Page Number를 추가하였습니다.

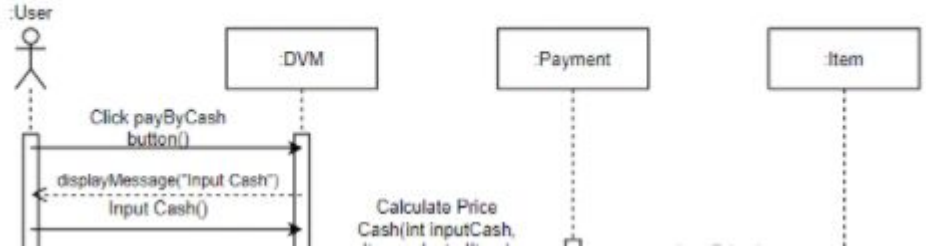
수정 요구사항

31	SDS	6	Interaction Diagram – 3.3.3	Calculate Price 에 '(Cash)'가 누락되어 추가 필요

요구사항 반영 후 문서 내용

: SDS Interaction Diagram 3.3.3

3.3.3. Pay by Cash & Calculate Price(Cash)



Structural Coverage

개선 요구사항 반영 결과

Element	Missed Instructions	Cov.	Missed Branches	Cov.
giveCode(Item)		55%		50%

```
213.    String giveCode(Item selectedItem) {
214.        char[] tmp = new char[6];
215.        for(int i=0; i<tmp.length; i++) {
216.            int div = (int) Math.floor( Math.random() * 2 );
217.
218.            if(div == 0) { // 0이면 숫자로
219.                tmp[i] = (char) (Math.random() * 10 + '0') ;
220.            }else { //1이면 알파벳
221.                tmp[i] = (char) (Math.random() * 26 + 'A') ;
222.            }
223.        }
224.        String code = new String(tmp);
225.        while (codeTable.keySet().contains(code)) {
226.            tmp = new char[6];
227.            for(int i=0; i<tmp.length; i++) {
228.                int div = (int) Math.floor( Math.random() * 2 );
229.
230.                if(div == 0) { // 0이면 숫자로
231.                    tmp[i] = (char) (Math.random() * 10 + '0') ;
232.                }else { //1이면 알파벳
233.                    tmp[i] = (char) (Math.random() * 26 + 'A') ;
234.                }
235.            }
236.            code = new String(tmp);
237.        }
238.        codeTable.put(code, selectedItem.getItemName());
239.        return code;
240.    }
241. }
```

```
String giveCode(Item selectedItem, Boolean payByCash) {
    char[] tmp;
    String code = "000000";
    while (code == "000000" || codeTable.keySet().contains(code)) {
        tmp = new char[6];
        for(int i=0; i<tmp.length; i++) {
            int div = (int) Math.floor( Math.random() * 2 );

            if(div == 0) { // 0이면 숫자로
                tmp[i] = (char) (Math.random() * 10 + '0') ;
            }else { //1이면 알파벳
                tmp[i] = (char) (Math.random() * 26 + 'A') ;
            }
        }
        code = new String(tmp);
    }
    codeTable.put(code, selectedItem.getItemName());
    if (payByCash) totalCash += selectedItem.getItemPrice();
    return code;
}
```

중복 제거를 위한 반복문에 접근하지 않아서 Test Coverage가 채워지지 않는 문제

-> 무조건 한 번은 반복문에 접근하도록 giveCode의 소스코드 수정

getRegion()	!	0%	n/a
getLocation()	!	0%	n/a

```
@Test
void getRegion() {
```

```
@Test
void getLocation() {
```

두 개의 get 함수에 대한 Test Case 부재
-> Test Case 생성

Brute Force Testing

개선 요구사항 반영 결과

수정 요구사항

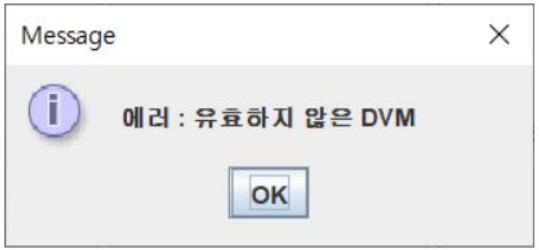
1	11 DVMs	자판기를 11 개 만드는 게 불가한지 확인한다.	Fail

요구사항 반영 결과

: 자판기를 11대 이상 생성하려고 시도할 경우 null값을 보유한 DVM 반환

→ 해당 DVM으로 GUI를 실행하면 오류 메시지 출력 후 GUI 자동 종료

```
자판기는 최대 10개까지만 생성 가능합니다.
[Another DVM 10]자판기는 생성되지 않았습니다.
```



수정 요구사항

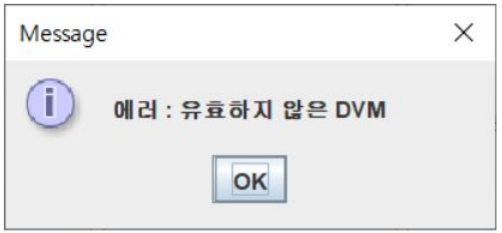
2	Show 20 and Sell 7 item	자판기가 20 개의 음료를 가지며 재고가 있는 음료가 7 개 존재함을 확인한다.	Fail

요구사항 반영 결과

: 자판기가 20개의 음료를 가지고 있지 않거나 재고가 있는 음료가 7개가 아닌 경우 null값을 보유한 DVM 반환 → 해당 DVM으로 GUI를 실행하면 오류 메시지 출력 후 GUI 자동 종료

```
아이템리스트는 반드시 20개의 아이템을 보유해야 합니다.
[Another DVM 1]자판기는 생성되지 않았습니다.
```

```
재고가 유효한 상품이 7개가 아닌 자판기는 생성할 수 없습니다.
[Another DVM 2]자판기는 생성되지 않았습니다.
```



수정 요구사항

10	Cash increase	관리자 메뉴에서 자판기의 현금보유량을 500 으로 설정한 뒤 가격이 1000 인 음료 A 를 구매했을 때, 현금보유량의 증가를 확인한다.	Fail

요구사항 반영 결과

: 최종적으로 상품, 쿠폰번호를 발행하는 함수인 **giveItem, giveCode**에 현금결제를 할 경우 **현금 보유량 (totalCash)을 상품 가격만큼 추가하는 소스코드 작성**

```
void giveItem(Item selectedItem, Boolean payByCash) {
    if (selectedItem.getItemAmount() > 0) {
        for (Item item: itemList) {
            if (item.getItemName().equals(selectedItem.getItemName())) {
                item.reduceAmount();
                if (payByCash) totalCash += item.getItemPrice();
            }
        }
    }
}
```

```
    codeTable.put(code, selectedItem.getItemName());
    if (payByCash) totalCash += selectedItem.getItemPrice();
    return code;
}
```

수정 요구사항

7	Left one item access - Code	음료 A 에 대한 재고가 없는 자판기 1 과 음료 A 에 대한 재고가 1 개 남은 자판기 2 를 설정한다. 자판기 1 에서 음료 A 에 대한 구매를 누른 뒤, 자판기 2 에서 음료 A 를 인증코드로 구매한다. 이후 자판기 1 에서 '다른 DVM 확인'을 입력하는 경우 주위에 가능한 자판기가 없다는 결과를 확인한다.	Fail

요구사항 반영 결과

: 해당 요구사항에 대해서는 현재 프로그램에서 재고가 존재하는 상품에 대한 인증코드 구매가 불가능하기 때문에 (재고 없는 상품에 대해서만 가능) 검증팀과 논의 후 Issue를 해결한 것으로 처리하였음

수정 요구사항

5	Left one item access - Cash	음료 A 에 대한 재고가 없는 자판기 1 과 음료 A 에 대한 재고가 1 개 남은 자판기 2 를 설정한다. 자판기 1 에서 음료 A 에 대한 구매를 누른 뒤, 자판기 2 에서 음료 A 를 현금으로 구매한다. 이 후 자판기 1 에서 '다른 DVM 확인'을 입력하는 경우 주위에 가능한 자판기가 없다는 결과를 확인한다.	Fail
6	Left one item access - Card	음료 A 에 대한 재고가 없는 자판기 1 과 음료 A 에 대한 재고가 1 개 남은 자판기 2 를 설정한다. 자판기 1 에서 음료 A 에 대한 구매를 누른 뒤, 자판기 2 에서 음료 A 를 카드로 구매한다. 이 후 자판기 1 에서 '다른 DVM 확인'을 입력하는 경우 주위에 가능한 자판기가 없다는 결과를 확인한다.	Fail

14	Item inventory check – Non-item	음료 A 에 대한 재고가 없는 자판기 1 과 음료 A 에 대한 재고가 1 개 남은 자판기 2 를 설정한다. 자판기 1 에서 음료 A 에 대한 구매를 누른 뒤, 자판기 2 의 관리자 메뉴에서 음료 A 에 대한 재고를 0 으로 수정한다. 이 후 자판기 1 에서	Fail
		'다른 DVM 확인'을 입력하는 경우 주위에 가능한 자판기가 없다는 결과를 확인한다.	
15	Item inventory check – Item	음료 A 에 대한 재고가 없는 자판기 1, 자판기 2 와 음료 A 에 대한 재고가 2 개 남은 자판기 3 을 설정한다. 자판기 1 에서 음료 A 에 대한 구매를 누른 뒤, 자판기 2 의 관리자 메뉴에서 음료 A 에 대한 재고를 1 으로 수정한다. 이 후 자판기 1 에서 '다른 DVM 확인'을 입력하는 경우 자판기 2 에 대한 거리 결과를 확인한다.	Fail

요구사항 반영 결과

: 해당 요구사항들에 대해서는 자체적으로 Testing을 수행한 결과 Pass되는 것으로 확인되어 검증팀과 논의 후 Issue를 해결한 것으로 처리하였음

Thank You.