KU 건국대학교 KONKUK UNIV.

2nd Testing

<system test & static analysis>

구상준, 류다은, 노현용, 김경희

KU 건국대학교 KONKUK UNIV.

Contents

1. Category Partition Test

Environments: 시스템 동작 흐름 관련 조건들

항목명	설명
Inventory 항목 수 검증	itemCount() 결과와 항목명 일치 여부 확인
유효/무효한 itemCode 처리	code == 0 또는 존재하지 않을 경우 nullptr 반환
재고 확인 메서드 isAvailable	수량 및 코드 유효성에 따른 bool 반환 확인
재고 차감 및 update 동작	유효한 경우 update 발생 / 실패 시 update 미호출 및 수량 유지
JSON 메시지 key/value 확인	req_prepay 포함 여부, itemCode 정확성 등 메시지 내용 확인
거리 계산 및 DVM 선택	calculateDistance 계산 및 getClosestDVM 선택 로직 검증
인증코드 생성 및 유효성 검사	인증코드 길이 및 문자 조건 확인, T/F 여부 처리
인증코드 사용 여부 처리	이미 사용된 코드인 경우 false 반환 / 저장 여부 확인
인증코드 오류 처리	존재하지 않는 코드 입력 시 (-1, -1) 반환
결제 처리 결과	성공 시 수량 차감 및 success 반환, 실패 시 수량 유지

Parameters: 사용자 입력 및 처리 데이터

항목명	설명
메뉴 선택 동작	사용자가 일반 구매 또는 사전결제 중 선택함. 이후 처리 흐름이 갈림.
재고 확인 로직	isAvailable() 호출로 재고 충분/부족 여부 판단. 부족 시 차감 불가.
재고 차감 처리	update() 호출로 재고 차감. 실패 시 차감되지 않음
사전결제 인증코드 생성	generatePrepayCode()로 인증코드 생성 후 저장
인증코드 유효성 검증	checkCodeAvailable() 등에서 사용 여부 확인 및 조건 검증
인증코드 기반 재고 연동	유효 코드 입력 시 itemCode, 수량 확인 후 제공 여부 결정
메시지 파싱 처리	req_prepay 포함 여부, itemCode 필드 등 JSON key 검사
타 자판기 선택 및 거리 비교	getClosestDVM()로 거리 계산 및 ID 우선 선택 처리
결제 결과 처리 흐름	정상 결제 시 재고 차감 / 잔액 부족 시 실패 메시지 반환
예외 처리 흐름	잘못된 입력값, 포맷 오류, 존재하지 않는 코드 등에 대한 처리 포함

테스트 코드

```
Environments:

  Menu Selection:
    Purchase.              [property Purchase]
    Prepayment.             [property Prepayment]
    Invalid Format.        [error]

  Stock Status:
    Sufficient.            [property Sufficient]
    Insufficient.          [property BroadcastRequired] [property Insufficient]

  Code Usage Status:
    Used.                  [error]
    Unused.                [property ValidCode]

  Code Registration:
    Registered.            [property CodeRegistered]
    Unregistered.          [error]

  Code Format:
    Valid Format.          [property ValidFormat]
    Invalid Format.        [error]

  Item Code Validity:
    Valid Item Code.       [property ValidItemCode]
    Invalid Item Code.     [error]

  Card Format:
    Valid Format.          [property ValidCard]
    No Hyphen.             [error]
    Wrong Format.          [error]
```

```
Quantity Request:
  Valid Quantity.        [property QtyValid]
  Over Quantity.         [error]
  Zero or Negative.      [error]

Message Type:
  Stock Request.         [property MsgStock]
  Prepay Request.        [property MsgPrepay]
  Invalid Message.       [error]

Input Code Format:
  5 Alphanumeric.        [property CodeFormatValid]
  Less Than 5 Chars.     [error]
  Special Characters.    [error]

Parameters:

Socket Response:
  Fast.                  [property FastResponse]
  Delayed.               [property SlowResponse]

Payment Gateway:
  Online.                [property Online]
  Offline.               [error]

JSON Format:
  Valid.                 [property JsonValid]
  Malformed.             [error]

Communication Timeout:
  Normal.                [property CommNormal]
  Timeout.               [error]
```

```
Message Type:
  Stock Request.         [property MsgStock]
  Prepay Request.        [property MsgPrepay]
  Invalid Message.       [error]

Input Code Format:
  5 Alphanumeric.        [property CodeFormatValid]
  Less Than 5 Chars.     [error]
  Special Characters.    [error]
```

Parameters:

Socket Response:
Fast. [property FastResponse]
Delayed. [property SlowResponse]

Payment Gateway:
Online. [property Online]
Offline. [error]

JSON Format:
Valid. [property JsonValid]
Malformed. [error]

Communication Timeout:
Normal. [property CommNormal]
Timeout. [error]

Payment Result:
Success. [property PaymentSuccess]
Insufficient Balance. [error]
Server Error. [error]

Prepaid Drink Availability:
Available. [property DrinkAvailable]
Unavailable. [error]

테스트 케이스

```
Test Case 1     <error>  (Key = 1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0)
  Inventory Count Check : Inventory item count mismatch
  Invalid Item Code Handling : <n/a>
  Stock Check with isAvailable : <n/a>
  Invalid Stock Code in isAvailable : <n/a>
  Stock Update on Success : <n/a>
  Stock Not Updated on Failure : <n/a>
  Stock Maintained if Update Fails : <n/a>
  JSON Message Field Check : <n/a>
  Distance Calculation Check : <n/a>
  DVM Selection Error : <n/a>
  Code Generation Format Error : <n/a>
  Code Format Validation : <n/a>
  Code Length Validation : <n/a>
  Used Code Detection : <n/a>
  Code Usage Update Failure : <n/a>
  Code Save Failure : <n/a>
  Code Lookup Error : <n/a>
  Payment Success Case : <n/a>
  Payment Failure Case : <n/a>
  Card Format Error : <n/a>

Test Case 2     <error>  (Key = 0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0)
  Inventory Count Check : <n/a>
  Invalid Item Code Handling : Invalid item code returns nullptr
  Stock Check with isAvailable : <n/a>
  Invalid Stock Code in isAvailable : <n/a>
  Stock Update on Success : <n/a>
  Stock Not Updated on Failure : <n/a>
  Stock Maintained if Update Fails : <n/a>
  JSON Message Field Check : <n/a>
  Distance Calculation Check : <n/a>
  DVM Selection Error : <n/a>
  Code Generation Format Error : <n/a>
  Code Format Validation : <n/a>
  Code Length Validation : <n/a>
  Used Code Detection : <n/a>
  Code Usage Update Failure : <n/a>
  Code Save Failure : <n/a>
  Code Lookup Error : <n/a>
  Payment Success Case : <n/a>
  Payment Failure Case : <n/a>
  Card Format Error : <n/a>
```

Test Case 3 <error> (Key = 0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0)
Inventory Count Check : <n/a>
Invalid Item Code Handling : <n/a>
Stock Check with isAvailable : Insufficient stock returns false
Invalid Stock Code in isAvailable : <n/a>
Stock Update on Success : <n/a>
Stock Not Updated on Failure : <n/a>
Stock Maintained if Update Fails : <n/a>
JSON Message Field Check : <n/a>
Distance Calculation Check : <n/a>
DVM Selection Error : <n/a>
Code Generation Format Error : <n/a>
Code Format Validation : <n/a>
Code Length Validation : <n/a>
Used Code Detection : <n/a>
Code Usage Update Failure : <n/a>
Code Save Failure : <n/a>
Code Lookup Error : <n/a>
Payment Success Case : <n/a>
Payment Failure Case : <n/a>
Card Format Error : <n/a>

Test Case 4 <error> (Key = 0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0)
Inventory Count Check : <n/a>
Invalid Item Code Handling : <n/a>
Stock Check with isAvailable : <n/a>
Invalid Stock Code in isAvailable : Invalid code returns false
Stock Update on Success : <n/a>
Stock Not Updated on Failure : <n/a>
Stock Maintained if Update Fails : <n/a>
JSON Message Field Check : <n/a>
Distance Calculation Check : <n/a>
DVM Selection Error : <n/a>
Code Generation Format Error : <n/a>
Code Format Validation : <n/a>
Code Length Validation : <n/a>
Used Code Detection : <n/a>
Code Usage Update Failure : <n/a>
Code Save Failure : <n/a>
Code Lookup Error : <n/a>
Payment Success Case : <n/a>
Payment Failure Case : <n/a>
Card Format Error : <n/a>

Test Case 5 <error> (Key = 0.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0)
Inventory Count Check : <n/a>
Invalid Item Code Handling : <n/a>
Stock Check with isAvailable : <n/a>
Invalid Stock Code in isAvailable : <n/a>
Stock Update on Success : Stock updates on success
Stock Not Updated on Failure : <n/a>
Stock Maintained if Update Fails : <n/a>
JSON Message Field Check : <n/a>
Distance Calculation Check : <n/a>
DVM Selection Error : <n/a>
Code Generation Format Error : <n/a>
Code Format Validation : <n/a>
Code Length Validation : <n/a>
Used Code Detection : <n/a>
Code Usage Update Failure : <n/a>
Code Save Failure : <n/a>
Code Lookup Error : <n/a>
Payment Success Case : <n/a>
Payment Failure Case : <n/a>
Card Format Error : <n/a>

Test Case 6 <error> (Key = 0.0.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0)
Inventory Count Check : <n/a>
Invalid Item Code Handling : <n/a>
Stock Check with isAvailable : <n/a>
Invalid Stock Code in isAvailable : <n/a>
Stock Update on Success : <n/a>
Stock Not Updated on Failure : Update not called on failure
Stock Maintained if Update Fails : <n/a>
JSON Message Field Check : <n/a>
Distance Calculation Check : <n/a>
DVM Selection Error : <n/a>
Code Generation Format Error : <n/a>
Code Format Validation : <n/a>
Code Length Validation : <n/a>
Used Code Detection : <n/a>
Code Usage Update Failure : <n/a>
Code Save Failure : <n/a>
Code Lookup Error : <n/a>
Payment Success Case : <n/a>
Payment Failure Case : <n/a>
Card Format Error : <n/a>

Test Case 7 <error>
Stock Maintained if Update Fails : Stock unchanged when update fails

Test Case 8 <error>
JSON Message Field Check : Missing or invalid req_prepay key

Test Case 9 <error>
Distance Calculation Check : Incorrect distance calculation logic

Test Case 10 <error>
DVM Selection Error : getClosestDVM returns wrong ID

Test Case 11 <error>
Code Generation Format Error : Invalid code format during generation

Test Case 12 <error>
Code Format Validation : Code fails alphanumeric check

Test Case 13 <error>
Code Length Validation : Code too short or too long

Test Case 14 <error>
Used Code Detection : Already used code returns false

Test Case 15 <error>
Code Usage Update Failure : Used code not marked correctly

Test Case 16 <error>
Code Save Failure : Failed to store newly issued code

Test Case 17 <error>
Code Lookup Error : Code not found returns (-1,-1)

Test Case 18 <error>
Payment Success Case : Stock decreased and success returned

Test Case 19 <error>
Payment Failure Case : Payment failed and stock unchanged

Test Case 20 <error>
Card Format Error : Card format invalid (missing hyphen)

테스트 흐름 작성

Test Case 1 – Invalid Item Code Handling
목적: 존재하지 않는 itemCode를 입력했을 때 시스템이 적절히 무효 처리하는지 확인
입력: itemCode = 0 또는 존재하지 않는 코드
흐름:
1. 사용자 또는 요청 메시지에서 잘못된 itemCode 전달
2. 시스템에서 getItemInfo(itemCode) 호출
3. 해당 코드가 존재하지 않으므로 nullptr 반환
이후 로직 중단 또는 예외 처리 수행 (예: "Invalid item" 메시지 반환)

Test Case 2 – Stock Check with isAvailable
목적: 재고가 충분한지 확인하는 메서드의 판단 정확도 확인
입력: itemCode는 유효하지만 재고 수량이 0
흐름:
1. isAvailable(itemCode) 호출
2. 내부적으로 item 존재 여부와 수량 확인
3. 재고가 없으므로 false 반환
구매 불가 메시지 출력, 선택 불가 처리

Test Case 3 – Stock Update on Success
목적: 결제 완료 후 재고 수량이 정확히 차감되는지 검증
입력: 유효한 itemCode, 재고 충분, 결제 성공
흐름:
1. 사용자가 결제 진행
2. 결제 성공 → update(itemCode) 호출
3. 수량 -1 처리
수량이 실제로 차감됨 (inventory[itemCode].count--)

Test Case 4 – JSON Message Field Check
목적: 메시지 포맷에 필요한 key/value가 누락 없이 포함됐는지 확인
입력: JSON 메시지에서 req_prepay 또는 itemCode 누락
흐름:
1. JSON 메시지 수신
2. hasKey("req_prepay") or get("itemCode") 시 null
메시지 처리 실패 / 오류 응답 전송

Test Case 5 – Used Code Detection
목적: 이미 사용된 인증코드를 다시 사용할 경우 거부 처리 여부 확인
입력: 사용된 인증 코드 입력
흐름:
1. 입력된 코드가 DB 또는 캐시에 존재하는지 확인
2. 존재하지만 상태가 "used"로 표시되어 있음
사용 불가 메시지 출력 / 결제 차단

Test Case 6 – Payment Failure Case
목적: 결제 실패 시 수량이 그대로 유지되는지 검증
입력: 결제 실패 (ex. 카드 오류, 서버 실패 등)
흐름:
1. processPayment() 호출
2. 결과: 실패 → update() 호출되지 않음
3. 재고 수량 유지
수량 그대로, 로그 또는 오류 메시지 출력

테스트 결과

TestCase	Result	설명
TestCase1	PASS	정상
TestCase2	PASS	정상
TestCase3	FAIL	DVM 연결 fail
TestCase4	FAIL	DVM 연결 fail
TestCase5	PASS	정상
TestCase6	FAIL	DVM 연결 fail

3/6

50%

1. Brute Force Test

TestNum	TestName	TestDescription	Expected Output
BF-01	RUC 01 초기메뉴 선택	콘솔에 선택지를 보여주고, 사용자가 유효한 숫자(1 또는 2)를 입력할 때까지 반복 입력을 받는다.	1이나 2가 아니면 잘못 입력했다는 메세지와 함께 다시 입력하도록 해야함
BF-02	RUC 01 초기메뉴 선택	각 번호에 맞게 입력했을 때 올바르게 출력값이 나오는지 확인	1번 입력했을 경우 음료 구매 화면으로, 2번 입력 했을 경우 선결제 화면으로 이동
BF-03	RUC 02 음료 종류 및 수량 선택	유효하지 않은 번호 입력 시 결과 확인	음료 이외 번호 입력 시 오류처리 및 재입력 요청
BF-04	RUC 02 음료 종류 및 수량 선택	입력 안 한 후 초과시간 지났을 때 반응 확인	초기화면으로 돌아가야 함
BF-05	RUC 02 음료 종류 및 수량 선택	숫자가 아닌 문자열을 넣었을 때 반응 확인	문자열이 들어왔으니 재입력 요청
BF-06	RUC 02 음료 종류 및 수량 선택	수량을 0 및 음수로 입력했을 때 반응 확인	0과 음수 수량은 유효하지 않으므로 재입력 요청,
BF-07	RUC 02 음료 종류 및 수량 선택	구매하고 싶은 수량은 자리수 제한이 있는지	02 256 과 같이 수량이 너무 큰 값이 입력될 경우 최대값 이하로 입력해달라고 메세지 출력, 최대값이 몇으로 되어 있는가
BF-08	RUC 02 음료 종류 및 수량 선택	자판기에서 판매 가능한 전체 음료 아이템 목록을 출력한다.	1부터 20까지의 메뉴 display
BF-09	RUC 03 카드 정보 입력과 확인	음료 가격이 알맞게 계산되는지 확인	음료 가격 * 수량으로 문제 없음
BF-10	RUC 03 카드 정보 입력과 확인	유효한 카드정보와 유효하지 않은 카드정보 입력 시 결과 확인	유효한 카드 정보 입력시 결제가 완료되며, 유효하지 않은 카드 정보는 다시 입력 요청
BF-11	RUC 03 카드 정보 입력과 확인	카드번호에 공백, 특수문자가 포함 된 경우	잘못된 형식으로 간주되어 재입력 요청
BF-12	RUC 04 음료 판매 처리	결제 후 판매 목록에 정상적으로 Sale 객체가 추가되는지 확인	판매 기록이 생성되고 저장됨
BF-13	RUC 04 음료 판매 처리(현재 자판기)	결제 완료 후 Enter 누를 경우와 다른 key를 눌렀을 때 확인	Enter를 입력하면 메인 화면으로 돌아감
BF-14	RUC 04 음료 판매 처리(현재 자판기)	음료 결제 후 자판기에서 해당 수량만큼 차감되는지 확인	해당 수량만큼 차감 됨
BF-15	RUC 03 다른 자판기 조회 및 위치 안내	현재 자판기 및 다른 자판기의 재고 상태를 확인하고 본인의 재고가 충분할 경우 판매한다.	재고가 충분하지 않다면 가장 가까운 다른 판매기의 위치를 출력 및 선결제 진행
BF-16	RUC 05 다른 자판기 조회 및 위치 안내	재고가 충분하지 않을 경우 다른 자판기를 알려주거나 재고가 없을 경우 not available 출력	다른 자판기의 위치나 not available 메세지 출력
BF-17	RUC 05 다른 자판기 조회 및 위치 안내	카드 정보 잘못 입력 시, 초기화면으로 돌아가는건 아닌지 확인	카드 정보 잘못 입력 시, 재입력 요청
BF-18	RUC 05 다른 자판기 조회 및 위치 안내	다른 자판기에도 재고가 없으면 어떤 메세지 나오는지	재고가 없어서 판매할 수 없다는 메세지 출력
BF-19	RUC 05 다른 자판기 조회 및 위치 안내	재고가 있는 다른 자판기 위치가 제대로 출력되는지	(3, 4) 위치의 자판기에서 판매가능하다는 식의 메세지 출력
BF-20	RUC 05 다른 자판기 조회 및 위치 안내	인증코드가 정상적으로 5자리로 생성되는지 확인	영소자 5자리 코드 생성
BF-21	RUC 05 다른 자판기 조회 및 위치 안내	구매 가능한 자판기 위치와 인증 코드 출력되는지	위치 (3, 4) 인증코드 73GJr과 같은 형식으로 출력
BF-22	RUC 06 선결제 구입	특정 DVM 대상의 선결제를 처리하고 인증 코드를 반환한다.	선결제 금액을 Sale 객체에 판매 아이템 업데이트 동시에 총 판매 금액을 해당 금액 만큼 증가시켜준다.
BF-23	RUC 07 선결제 음료 수령	유효한 인증코드와 유효하지 않은 코드 입력 시 결과 확인	유효한 인증코드만 허가되어 음료 수령 가능
BF-24	RUC 07 선결제 음료 수령	음료 수령 시, 어떤 자판기에서 수령이 빠져나가는지 확인	선결제한 자판기에서 음료 수령이 차감 됨
BF-25	RUC 07 선결제 음료 수령	음료 제공 후 메인 화면으로 돌아갈 때 Enter가 아닌 다른 키가 입력했을 경우 확인	Enter를 입력하라는 메세지 출력
BF-26	RUC 07 선결제 음료 수령	인증코드를 입력받아 선결제된 음료를 수령 처리	sales 내에 일치하는 certCode가 있는 경우 수령 처리
BF-27	RUC 07 선결제 음료 수령	인증 코드를 한 번만 사용할 수 있도록 사용 여부 관리를 한다.	사용 여부를 boolean으로 잘 저장하고 있어야 함
BF-28	RUC 07 선결제 음료 수령	입력된 인증코드가 일치하고, 아직 사용되지 않았다면 사용 처리를 한다.	사용 처리 후 isUsed true로 업데이트

TestNum	pass/fail	비고
BF-01	Pass	
BF-02	Pass	
BF-03	Pass	
BF-04	Fail	입력시간 초과 되어도 돌아가지 않음
BF-05	Pass	
BF-06	Pass	
BF-07	Pass	
BF-08	Pass	
BF-09	Pass	
BF-10	Pass	
BF-11	Pass	재입력요청은 아닌 초기화면으로 돌아감
BF-12	Fail	Connect Error 뜨면서 확인 불가 및 본인 IP 노출 중
BF-13	Pass	
BF-14	Pass	
BF-15	Fail	Connect Error 뜨면서 확인 불가 및 본인 IP 노출 중
BF-16	Fail	Connect Error 뜨면서 확인 불가 및 본인 IP 노출 중
BF-17	Fail	Connect Error 뜨면서 확인 불가 및 본인 IP 노출 중
BF-18	Fail	Connect Error 뜨면서 확인 불가 및 본인 IP 노출 중
BF-19	Fail	Connect Error 뜨면서 확인 불가 및 본인 IP 노출 중
BF-20	Fail	Connect Error 뜨면서 확인 불가 및 본인 IP 노출 중
BF-21	Fail	Connect Error 뜨면서 확인 불가 및 본인 IP 노출 중
BF-22	Fail	Connect Error 뜨면서 확인 불가 및 본인 IP 노출 중
BF-23	Fail	Connect Error 뜨면서 확인 불가 및 본인 IP 노출 중
BF-24	Fail	Connect Error 뜨면서 확인 불가 및 본인 IP 노출 중
BF-25	Fail	Connect Error 뜨면서 확인 불가 및 본인 IP 노출 중
BF-26	Fail	Connect Error 뜨면서 확인 불가 및 본인 IP 노출 중
BF-27	Fail	Connect Error 뜨면서 확인 불가 및 본인 IP 노출 중
BF-28	Fail	Connect Error 뜨면서 확인 불가 및 본인 IP 노출 중

Pass rate = 12/28 = 42%