

2nd Testing

<system test & static analysis>

구상준, 류다은, 노현용, 김경희

Contents

1. Category Partition Test

1. Brute Force Test

1. Sonar Cloud

1. Category Partition Test

Environments: 시스템 동작 흐름 관련 조건들

테스트 코드 기반 조건	설명
재고 수량 비교	Inventory 클래스 내 isAvailable 메서드로 재고 수량 확인
재고 차감 처리	Inventory 클래스 내 update 메서드를 통해 수량 차감 여부 확인
음료 가격 확인	Beverage 객체의 getPrice 호출로 가격 정보 확인
아이템 존재 여부 확인	getItem을 통해 itemCode 유효성 검사
유효/무효 itemCode 분기 처리	존재하지 않는 itemCode에 대해 nullptr 처리
결제 시스템 응답 처리	FakePaymentSystem에서 사전 설정된 응답값 기반 결제 성공/실패 판단
선결제 코드 생성 처리	PrepaymentHandler에서 인증 코드 생성 및 저장 여부 검증
선결제 코드 중복 처리	중복 코드 저장 시 예외 혹은 false 반환 여부 확인
Broadcast 호출 여부	BroadCastImpl 또는 FakeBroadCast 통해 응답 전파 여부 확인
정상/비정상 흐름 분기	입력값 조작하여 성공 흐름과 오류 흐름 분기 처리 여부 검증

Parameters: 사용자 입력 및 처리 데이터

항목명	설명
Socket Response	소켓 응답이 빠르게 돌아올 경우 정상 처리
Socket Response	소켓 응답이 타임아웃되어 실패 처리
Payment Gateway	결제 시스템이 온라인 상태일 경우 정상 처리
Payment Gateway	결제 시스템이 오프라인일 경우 실패 처리
JSON Format	JSON 형식이 올바른 경우 정상 처리
JSON Format	JSON 형식이 잘못된 경우 오류 발생
Communication Timeout	통신이 정상 시간 내 완료된 경우
Communication Timeout	통신 지연으로 타임아웃 발생 시 실패 처리
Payment Result	결제가 정상적으로 완료된 경우
Payment Result	잔액 부족으로 결제 실패한 경우
Payment Result	결제 서버 오류로 인한 실패
Prepaid Drink Availability	사전 등록된 음료가 있는 경우 정상 처리
Prepaid Drink Availability	사전 등록된 음료가 없는 경우 오류

테스트 코드

Environments:

Menu Selection:

Purchase.	[property Purchase]
Prepayment.	[property Prepayment]
Invalid Format.	[error]

Stock Status:

Sufficient.	[property Sufficient]
Insufficient.	[property Insufficient] [property BroadcastRequired]

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.	[property ValidItemCode]
Invalid.	[error]

Card Format:

Valid Format.	[property ValidCard]
No Hyphen.	[error]
Invalid Format.	[error]

Message Token Presence:

Present.	[property TokenPresent]
Missing.	[error]

Inventory Capacity:

Within Limit.	[property InvAcceptable]
Exceeded.	[error]

JSON Fields:

Valid.	[property JsonValid]
Missing req_prepay.	[error]
Missing itemCode.	[error]

Message Load:
 Normal Load. [property LoadAcceptable]
 Overload. [error]

Concurrency:
 Single Access. [property AccessSafe]
 Concurrent Write. [error]

Parameters:

Socket Response:
 Fast. [property FastResponse]
 Timeout. [error]

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>
Stock Maintained if Update Fails : Stock unchanged when update fails
- Test Case 2 <error>
JSON Message Field Check : Missing or invalid req_prepay key
- Test Case 3 <error>
Distance Calculation Check : Incorrect distance calculation logic
- Test Case 4 <error>
DVM Selection Error : getClosestDVM returns wrong ID
- Test Case 5 <error>
Code Generation Format Error : Invalid code format during generation
- Test Case 6 <error>
Code Format Validation : Code fails alphanumeric check
- Test Case 7 <error>
Code Length Validation : Code too short or too long
- Test Case 8 <error>
Used Code Detection : Already used code returns false
- Test Case 9 <error>
Code Usage Update Failure : Used code not marked correctly
- Test Case 10 <error>
Code Save Failure : Failed to store newly issued code
|
- Test Case 11 <error>
Code Lookup Error : Code not found returns (-1,-1)
- Test Case 12 <success>
Payment Success Case : Stock decreased and success returned
- Test Case 13 <error>
Payment Failure Case : Payment failed and stock unchanged

Test Case 14 <success> (Key = 1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0)

Used Code Detection : Already used code returns false

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 15 <error> (Key = 0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.0)

Used Code Detection : <n/a>

Code Usage Update Failure : Used code not marked correctly

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 16 <error> (Key = 0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0)

Used Code Detection : <n/a>

Code Usage Update Failure : <n/a>

Code Save Failure : Failed to store newly issued code

Code Lookup Error : <n/a>

Payment Success Case : <n/a>

Payment Failure Case : <n/a>

Card Format Error : <n/a>

Test Case 17 <error> (Key = 0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0.0)

Used Code Detection : <n/a>

Code Usage Update Failure : <n/a>

Code Save Failure : <n/a>

Code Lookup Error : Code not found returns (-1,-1)

Payment Success Case : <n/a>

Payment Failure Case : <n/a>

Card Format Error : <n/a>

Test Case 18 <success> (Key = 0.0.0.0.1.0.0.0.0.0.0.0.0.0.0.0.0)

Used Code Detection : <n/a>

Code Usage Update Failure : <n/a>

Code Save Failure : <n/a>

Code Lookup Error : <n/a>

Payment Success Case : Stock decreased and success returned

Payment Failure Case : <n/a>

Card Format Error : <n/a>

Test Case 19 <error> (Key = 0.0.0.0.0.1.0.0.0.0.0.0.0.0.0.0)

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 : Payment failed and stock unchanged

Card Format Error : <n/a>

Test Case 20 <error> (Key = 0.0.0.0.0.1.0.0.0.0.0.0.0.0.0.0)

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 : Card format invalid (missing hyphen)

테스트 흐름 작성

Test Case 14 – Card Format Error

목적: 카드 입력값의 형식이 하이픈 누락 등으로 유효하지 않은 경우, 시스템이 적절히 처리하는지 확인

입력: 잘못된 카드 번호 (예: 하이픈 없이 1234567812345678)

흐름:

1. 사용자가 카드 번호 입력
2. 카드 포맷 검증 (regex 또는 split 기준)
3. 하이픈이 없거나 잘못된 형식 → 에러 메시지 출력

Test Case 15 – Invalid Item Code

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 않으면 null 반환
- 이후 요청 중단 또는 에러 메시지 처리 수행 (예: 'Invalid item' 메시지 반환)

Test Case 16 – Inventory Overflow

목적: 인벤토리에 허용된 범위 이상으로 아이템이 추가될 때 예외 발생 여부 확인

입력: 테스트를 위한 대량 아이템 등록 (예: 1000개 이상)

흐름:

1. 반복적으로 Inventory::addItem() 호출
2. 내부적으로 배열 또는 리스트 용량 초과
3. 예외 또는 삽입 실패 반환

Test Case 17 – Valid Code Registration

목적: 유효한 코드가 정상적으로 등록되고 사용 가능한 상태로 전환되는지 확인

입력: 형식이 올바르고 등록되지 않은 인증 코드

흐름:

1. 사용자가 선결제 인증 코드 입력
2. 코드 유효성 확인 (형식, 중복 여부)
3. 인증 코드 저장 및 사용 가능 상태 전환

Test Case 18 – Missing Menu Token

목적: JSON 요청 메시지에 필수 키(menu)가 누락되었을 때 적절한 오류 처리를 수행하는지 확인

입력: menu 키가 없는 JSON 메시지

흐름:

1. JSON 메시지 수신
2. hasKey("menu") 또는 get("menu") 시 null
3. 필수 필드 누락 → 오류 메시지 반환 또는 예외 처리

Test Case 19 – Message Overload

목적: 짧은 시간 내에 너무 많은 메시지가 도착했을 때 시스템이 과부하를 감지하고 처리하는지 확인

입력: 동시 접속 또는 메시지 폭주 (e.g., 100개 이상)

흐름:

1. 동시에 여러 메시지 수신
2. 처리 큐 또는 스레드풀 초과
3. 일부 요청 무시 또는 에러 반환 ("System overload")

Test Case 20 – Concurrent Access Conflict

목적: 여러 스레드가 동시에 동일한 파일 또는 데이터에 접근할 때 충돌을 감지하고 처리하는지 확인

입력: 동일한 인증코드 저장 요청이 동시에 여러 스레드에서 발생

흐름:

1. 동시에 동일한 코드로 파일 쓰기 요청 발생
2. 락(lock) 미사용 시 race condition 발생
3. 충돌 발생 → 하나의 요청만 성공, 나머지는 에러

테스트 결과

Test Case	Title	Result	Reason
TestCase 14	Card Format Error	Fail	하이픈 누락으로 인해 카드 형식이 유효하지 않음
TestCase 15	Invalid Item Code	Fail	등록되지 않았거나 잘못된 형식의 itemCode
TestCase 16	Inventory Overflow	Fail	등록된 아이템 수가 인벤토리 허용치를 초과함
TestCase 17	Valid Code Registration	Pass	유효한 코드가 정상적으로 저장되고 사용 가능 상태로 전환됨
TestCase 18	Missing Menu Token	Fail	JSON 요청에 menu 필드가 누락되어 잘못된 요청으로 처리됨
TestCase 19	Message Overload	Fail	동시 메시지 과부하로 시스템이 처리 불가 상태에 도달함
TestCase 20	Concurrent Access Conflict	Fail	동일한 자원에 여러 스레드가 접근하면서 충돌 발생

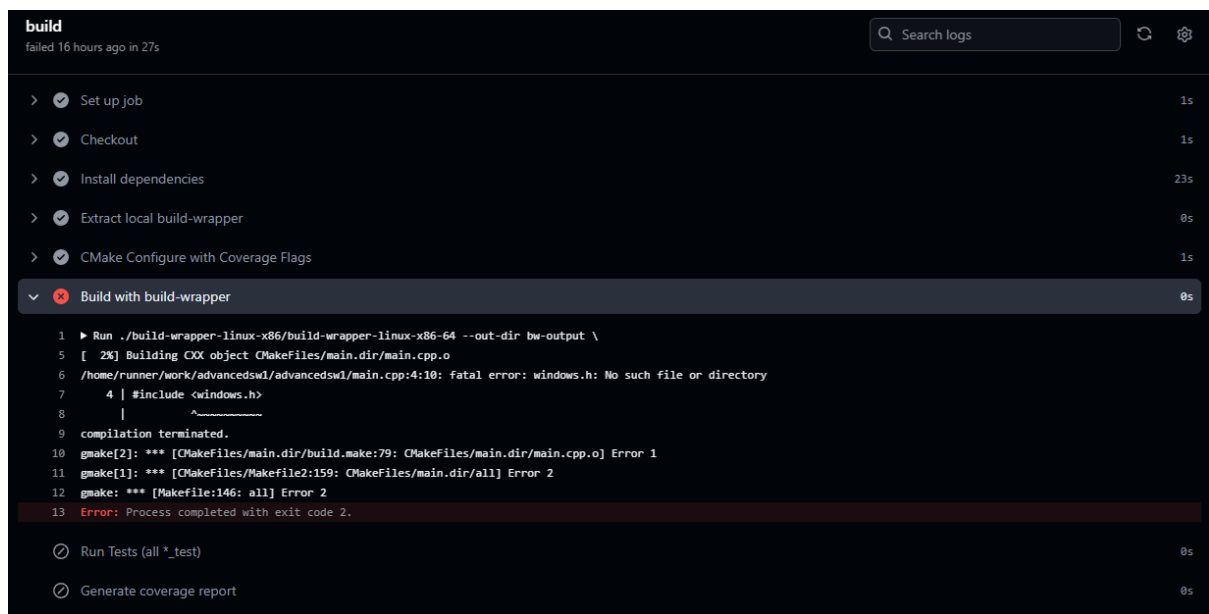
1 Brute Force Test

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 메시지와 함께 다시 입력하도록 해야함
BF-02	RUC 01 초기메뉴 선택	각 번호에 맞게 입력했을 때 올바르게 출력값이 나오는지 확인	1번 입력했을 경우 종료 구매 화면으로, 2번 입력했을 경우 선결제 화면으로 이동
BF-03	RUC 02 음료 종류 및 수량 선택	유효하지 않은 번호 입력 시 결과 확인	음료 이외의 번호 입력 시 오류처리 및 재입력 요청
BF-04	RUC 02 음료 종류 및 수량 선택	입력 안 함 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 자리가 수 제한이 있는지	큰 값이 입력될 경우 최대값 이하로 입력해달라고 메시지 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 잘못게 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 가격이 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 저장되는지 확인	해당 수량만큼 저장 됨
BF-15	RUC 03 다른 자판기 조회 및 위치 안내	현재 자판기 및 다른 자판기의 재고 상태를 확인하고 본인의 재고가 충분한 경우 판매한다.	재고가 충분하지 않다면 가장 가까운 다른 판매기의 위치를 출력 및 선결제 진행
BF-16	RUC 05 다른 자판기 조회 및 위치 안내	본인의 재고가 충분하지 않을 경우 다른 자판기의 위치를 알려주거나 not available을 출력한다.	다른 자판기의 위치가 not available 메시지 출력
BF-17	RUC 05 다른 자판기 조회 및 위치 안내	카드 정보 잘못 입력 시, 초기 화면으로 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 메시지가 나오는지	재고가 없어서 판매할 수 없다는 메시지 출력
BF-19	RUC 05 다른 자판기 조회 및 위치 안내	재고가 있는 다른 자판기 위치가 제대로 출력되는지	(3, 4) 위치의 자판기에서 판매가능하다는 식의 메시지 출력
BF-20	RUC 05 다른 자판기 조회 및 위치 안내	인종코드가 정상적으로 5자리로 생성되는지 확인	영숫자 5자리 코드 생성
BF-21	RUC 05 다른 자판기 조회 및 위치 안내	구매 가능한 자판기 위치와 인종 코드 출력되는지	위치 (3, 4) 인종코드 73GJR과 같은 형식으로 출력
BF-22	RUC 06 선결제 구입	특정 DVM 대상의 선결제를 처리하고 인종 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 인종코드와 유효하지 않은 코드 입력 시 결과 확인	유효한 인종코드만 허가되어 음료 수령 가능
BF-24	RUC 07 선결제 음료 수령	음료 수령 시, 어떤 자판기에서 수령이 빠져나가는지 확인	선결제한 자판기에서 음료 수령이 저장 됨
BF-25	RUC 07 선결제 음료 수령	음료 제공 후 메인 화면으로 돌아갈 때 Enter가 아닌 다른 키가 입력했을 경우 확인	Enter를 입력하라는 메시지 출력
BF-26	RUC 07 선결제 음료 수령	인종코드를 입력받아 선결제된 음료를 수령 처리	sales 내에 일치하는 centCode가 있는 경우 수형 처리
BF-27	RUC 07 선결제 음료 수령	인종 코드를 한 번만 사용할 수 있도록 사용 여부 관리를 한다.	사용 여부를 boolean으로 잘 저장하고 있어야 함
BF-28	RUC 07 선결제 음료 수령	입력한 인종코드가 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	pass	
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	pass	
BF-13	pass	
BF-14	pass	
BF-15	pass	
BF-16	pass	
BF-17	pass	
BF-18	pass	
BF-19	pass	
BF-20	pass	
BF-21	pass	
BF-22	pass	
BF-23	pass	
BF-24	pass	
BF-25	pass	
BF-26	pass	
BF-27	pass	
BF-28	pass	

Pass rate = 100%

1. Sonar Cloud



The screenshot shows a SonarCloud build log for a project named 'build'. The build failed 16 hours ago in 27s. The log lists several steps: 'Set up job' (1s), 'Checkout' (1s), 'Install dependencies' (23s), 'Extract local build-wrapper' (0s), and 'CMake Configure with Coverage Flags' (1s). The final step, 'Build with build-wrapper' (0s), failed. The error message is: 'Error: Process completed with exit code 2.' The log also shows the following output: '1 Run ./build-wrapper-linux-x86/build-wrapper-linux-x86-64 --out-dir bw-output \', '5 [2%] Building CXX object CMakeFiles/main.dir/main.cpp.o', '6 /home/runner/work/advancedsw1/advancedsw1/main.cpp:4:10: fatal error: windows.h: No such file or directory', '7 4 | #include <windows.h>', '8 |', '9 compilation terminated.', '10 gmake[2]: *** [CMakeFiles/main.dir/build.make:79: CMakeFiles/main.dir/main.cpp.o] Error 1', '11 gmake[1]: *** [CMakeFiles/Makefile2:159: CMakeFiles/main.dir/all] Error 2', '12 gmake: *** [Makefile:146: all] Error 2', '13 Error: Process completed with exit code 2.'

```
build
failed 16 hours ago in 27s

> Set up job 1s
> Checkout 1s
> Install dependencies 23s
> Extract local build-wrapper 0s
> CMake Configure with Coverage Flags 1s
v Build with build-wrapper 0s

1 Run ./build-wrapper-linux-x86/build-wrapper-linux-x86-64 --out-dir bw-output \
5 [ 2%] Building CXX object CMakeFiles/main.dir/main.cpp.o
6 /home/runner/work/advancedsw1/advancedsw1/main.cpp:4:10: fatal error: windows.h: No such file or directory
7 4 | #include <windows.h>
8 |
9 compilation terminated.
10 gmake[2]: *** [CMakeFiles/main.dir/build.make:79: CMakeFiles/main.dir/main.cpp.o] Error 1
11 gmake[1]: *** [CMakeFiles/Makefile2:159: CMakeFiles/main.dir/all] Error 2
12 gmake: *** [Makefile:146: all] Error 2
13 Error: Process completed with exit code 2.

Run Tests (all *_test) 0s
Generate coverage report 0s
```

Windows 개발 환경에서 만들어져서 Static Analysis 수행 할 수 없음.

Ubuntu 버전으로 소스 코드 변경해주길 기다리는 중.