Software Verification & Validation
System Test & Static Analysis Report
[Team A1] 2nd

Team 1

201711381 김소현

201711394 민하은

201711401 염혜지

201711407 이가현

201711435 홍예주

Index

1. Specification Review

A. Stage 2030 planning

Category	Number	Review
2131	1	use case 'DVM을 선택한다'가 들어갈 필요가 없다. stage 2131-2가 DVM을 선택하는 use case이기 때문이다.
2131	3	Ref 1.2 'DVM을 선택한다'는 stage 2131-2에 해당하기 때문에 지워야 한다. Ref 3.1 '결제 방법 중 하나를 선택한다'는 use case stage 2131-5에 해당하므로 지워야 한다.
2131	4	Ref 1.2, 6.1를 지워야 한다. Ref 6.1의 경우 use case 12번에 해당한다.
2131	6	ref 6.1 '선결제를 진행한다'는 지워야 한다. Ref 6.1의 경우 use case 12번에 해당한다. event 진행 과정과 선결제는 무관하다.

General Review

-> 전체적으로 시스템이 어떻게 진행되는지 알기 쉬워졌다.

-> 뒤로가기 기능이 없다. 중간에 결제 취소등을 사용자가 자의로 할 수 있어야 한다.

2. Category – partition Testing Report

A. Test case

Group	Category	Value	constraint	key
DVM_number	select DVM number	1>= and <=8		1
		<1	error	2
		>8	error	3
		1 to 8 with a leading zero		4
		not (1 to 8) with a leading zero	error	5
Drink	select Drink	1~7 with stock	property cs	1
		8~20 with stock	property cns	2

		<1 or >20	error	3
Payment_select	select Payment	1	if cs | property card	1
		1 next 2	if cs | property code	2
		first 2	error	3
		exclude 1,2	error	4
Payment_process	card payment (선결제가 아닌 카드결제)	valid, 잔고ㅇ	if card | property card_s	1
		valid, 잔고 x	if card	2
		8자리인데 invalid	error	3
		8자리가 아닌경우	error	4
	pre-payment (선결제인데 카드결제)	valid, 잔고ㅇ	if cns | property pre	1
		valid, 잔고 x	if cns	2
		8자리인데 invalid	error	3
		8자리가 아닌경우	error	4
	get_code	get valid code	if pre	1
		get invalid code	error	2
		get no code	error	3
	code payment (선결제하고나서 코드결제)	valid한 코드	if code | property code_s	1
		5자리의 invalid코드	error	2
		5자리가 아닌 경우	error	3
Environment	number of DVM	8		1
		not 8	error	2
	number of Drink types	20		1
		not 20	error	2
	DVM drink stock	not zero		1
		both zero	single	2
Drink_output	drink_output	card valid ouput	if card_s	1
		code valid output	if code_s	2
		invalid output	error	3

➔ 5 x 3 x 4 x 4 x 4 x 3 x 3 x 2 x 2 x 2 x 3 = 207360 개

1. single constraint 적용

→ 103681개로 감소

2. error constraint 적용

→ 82개로 감소

3. property constraint 적용

→ 28로 감소

B. Test Result

번호	expected output	test result	비고
1	전체 DVM 수가 8개	pass	
2	전체 음료 종류가 20개	pass	
3	모든 DVM에 재고가 없으면 재고가 없다고 알려줘야 함	pass	
4	DVM의 번호가 1 미만인 것을 고를 수 없음	pass	
5	DVM의 번호가 8 초과인 것을 고를 수 없음	pass	
6	0으로 시작하는 1과 8 사이의 숫자가 아닌 DVM 번호를 고를 수 없음	pass	
7	음료 번호를 1 미만이나 20 초과로 고를 수 없음	pass	
8	선결제를 하지 않고 코드 결제를 할 수 없음	pass	
9	결제 선택에서 1과 2를 제외한 숫자를 고를 수 없음	pass	
10	카드 번호가 8자리라도 유효하지 않으면 결제할 수 없음	pass	
11	카드 결제 시 카드 번호가 8자리가 아니면 결제할 수 없음	pass	
12	선 결제 시 카드번호가 8자리라도 유효하지 않으면 결제할 수 없음	pass	
13	선결제 시 카드번호가 8자리가 아니면 결제할 수 없음	pass	
14	선결제 시 유효하지 않은 코드가 나오면 안됨	pass	
15	선결제 시 코드가 나오지 않으면 안됨.	pass	
16	코드 결제 시 코드가 5자리지만 유효한 코드 아니면 결제가 되지 않음	pass	
17	코드 결제 시 코드가 5자리가 아니면 결제가 되지 않음	pass	
18	음료가 유효하지 않게 출력되면 안됨	pass	
19	잔액이 없는 카드로 결제를 시도해서 실	pass	

	패함.		
20	잔액이 있는 카드로 올바르게 결제 진행하여 올바른 음료가 나옴.	fail	구매한 dvm의 번호가 잘 못나옴
21	선결제 후 올바른 코드로 결제하여 올바른 음료가 나옴.	fail	구매한 dvm의 번호가 잘 못나옴
22	잔액이 없는 카드로 선결제를 시도하여 선결제 실패함.	pass	
23	선결제는 성공하여 올바른 코드 5자리를 받음.	pass	
24	재고가 있는 음료를 선택함. 잔고가 없는 유효한 8자리 카드번호를 입력하여 카드결제에 실패함.	pass	
25	재고가 있는 음료를 선택함. 잔고가 있는 유효한 8자리 카드번호를 입력하여, 카드결제에 성공함.	fail	구매한 dvm의 번호가 잘 못나옴
26	먼저 선결제를 완료함. 0으로 시작하는 DVM 번호를 입력 후, 재고가 있는 음료를 선택함. 유효한 코드번호를 입력하여, 코드결제에 성공함.	fail	구매한 dvm의 번호가 잘 못나옴
27	0으로 시작하는 DVM 번호를 입력 후, 재고가 없는 음료를 선택함. 잔고가 없는 유효한 8자리 카드번호를 입력하여, 코드를 받지 못함(선결제 실패).	pass	
28	0으로 시작하는 DVM 번호를 입력 후, 재고가 없는 음료를 선택함. 잔고가 있는 유효한 8자리 카드번호를 입력하여, 코드를 받음(선결제 성공).	pass	

Test Result = 24/28 = 대략 85.71%

3. Brute Force Testing Report

A. Test Case

Test	Test Num	Description
Select drink test	1-1	선택한 음료로 결제 진행이 잘 되는지 확인한다.
	1-2	없는 번호의 음료를 선택한다.
	1-3	정해진 가격대로 정확히 출력되는지 확인한다.
Proceed Card-Payment test	2-1	사용 가능한 카드를 입력한다.
	2-2	사용 불가능한 카드를 입력한다.
	2-3	카드의 잔액이 충분할 때만 결제가 진행되는지 확인한다.
	2-4	카드의 잔액이 부족할 경우 잔액 부족 메시지를 출력하는지 확인한다.
Check current machine stock test	3-1	현재 자판기의 재고가 정확한지 확인한다.
	3-2	현재 자판기에서 재고가 0개일 때 재고가 부족하다는 메시지를 출력하는지 확인한다.
Check other DVMs stock test	4-1	자기 외의 다른 자판기의 재고가 정확하게 표시되는지 확인한다.
DVMs location test	5-1	재고가 있는 자판기의 위치를 올바르게 출력하는지 확인한다.
Make code test	6-1	랜덤으로 생성된 인증코드가 기존의 인증코드와 중복되지 않는지 확인한다.
	6-2	화면에 출력되는 인증코드가 사용자가 선결제해서 생성된 인증코드와 일치하는지 확인한다.
Enter code test	7-1	코드를 생성한 후 생성한 코드와 같은 코드를 입력해본다.
	7-2	코드를 생성한 후 생성한 코드와 다른 코드를 입력해본다.
	7-3	생성한 코드가 없는 채로 코드 입력을 시도해본다.
Check code test	8-1	유효하지 않은 인증코드를 입력해본다.
	8-2	이미 사용한 인증코드를 입력해본다.
Check drink test	9-1	제공할 음료가 사용자가 결제한 음료와 일치하는지 확인한다.
Provide drink test	10-1	음료가 정확히 사용자에게 제공되는지 확인한다.
Check pay result	11-1	코드 결제 후 음료 구매 완료에 대한 메시지가 올바르게 출력되는지 확인한다.

B. Test Result

No	P/F	Expected Output	Test result
1-1	P	선택한 음료에 대한 결제가 성공적으로 진행된다.	선택한 음료에 대한 결제가 성공적으로 진행된다.
1-2	P	잘못된 번호입니다 라는 메시지가 출력된다.	잘못된 번호입니다 라는 메시지가 출력된다.
1-3	P	선택한 음료에 대한 가격 정확히 출력된다.	선택한 음료에 대한 가격 정확히 출력된다.
2-1	P	카드결제가 성공적으로 진행된다.	카드결제가 성공적으로 진행된다.
2-2	P	에러 메세지가 뜬 후 초기화면으로 돌아간다.	에러 메세지가 뜬 후 초기화면으로 돌아간다.
2-3	P	카드결제가 성공적으로 진행되고 음료가격만큼 잔액이 차감된다.	카드결제가 성공적으로 진행되고 음료가격만큼 잔액이 차감된다.
2-4	P	잔액부족 메시지를 출력한다.	잔액부족 메시지를 출력한다.
3-1	P	dvm의 음료가 재고 개수만큼 구매된다.	dvm의 음료가 재고 개수만큼 구매된다.
3-2	P	재고없음 문구가 뜨고 초기화면으로 돌아간다.	재고없음 문구가 뜨고 초기화면으로 돌아간다.
4-1	P	다른 dvm의 음료가 재고 개수만큼 선결제 된다.	다른 dvm의 음료가 재고 개수만큼 선결제 된다.
5-1	P	DVM 번호 옆에 주소가 101 202 형태로 뜬다	DVM 번호 옆에 주소가 101 202 형태로 뜬다
6-1	P	메세지로 출력되는 인증코드가 이전과 겹치지 않는다.	메세지로 출력되는 인증코드가 이전과 겹치지 않는다.
6-2	P	선결제 후 출력되는 인증코드 번호와 코드결제 후 출력되는 인증코드 번호가 동일하다.	선결제 후 출력되는 인증코드 번호와 코드결제 후 출력되는 인증코드 번호가 동일하다.
7-1	P	코드결제가 성공적으로 진행된다.	코드결제가 성공적으로 진행된다.
7-2	P	유효하지 않는 인증코드입니다 라는 메시지가 출력된 후 초기화면으로 돌아간다.	유효하지 않는 인증코드입니다 라는 메시지가 출력된 후 초기화면으로 돌아간다.
7-3	P	유효하지 않는 인증코드입니다 라는 메시지가 출력된 후 초기화면으로 돌아간다.	유효하지 않는 인증코드입니다 라는 메시지가 출력된 후 초기화면으로 돌아간다.
8-1	P	유효하지 않는 인증코드입니다 라는 메시지가 출력된 후 초기화면으로 돌아간다.	유효하지 않는 인증코드입니다 라는 메시지가 출력된 후 초기화면으로 돌아간다.
8-2	P	유효하지 않는 인증코드입니다 라는 메시지가 출력된 후 초기화면으로 돌아간	유효하지 않는 인증코드입니다 라는 메시지가 출력된 후 초기화면으로 돌아간

		다.	다.
9-1	P	Dvm에서 선택한 음료와 구매 후 출력되는 음료가 같다.	Dvm에서 선택한 음료와 구매 후 출력되는 음료가 같다.
10-1	P	Dvm에서 선택한 음료와 구매 후 출력되는 음료가 같다.	Dvm에서 선택한 음료와 구매 후 출력되는 음료가 같다.
11-1	F	음료를 구매한 dvm 번호와 구매완료 메세지로 출력된 dvm번호가 같아야한다	Dvm번호가 같지 않다.

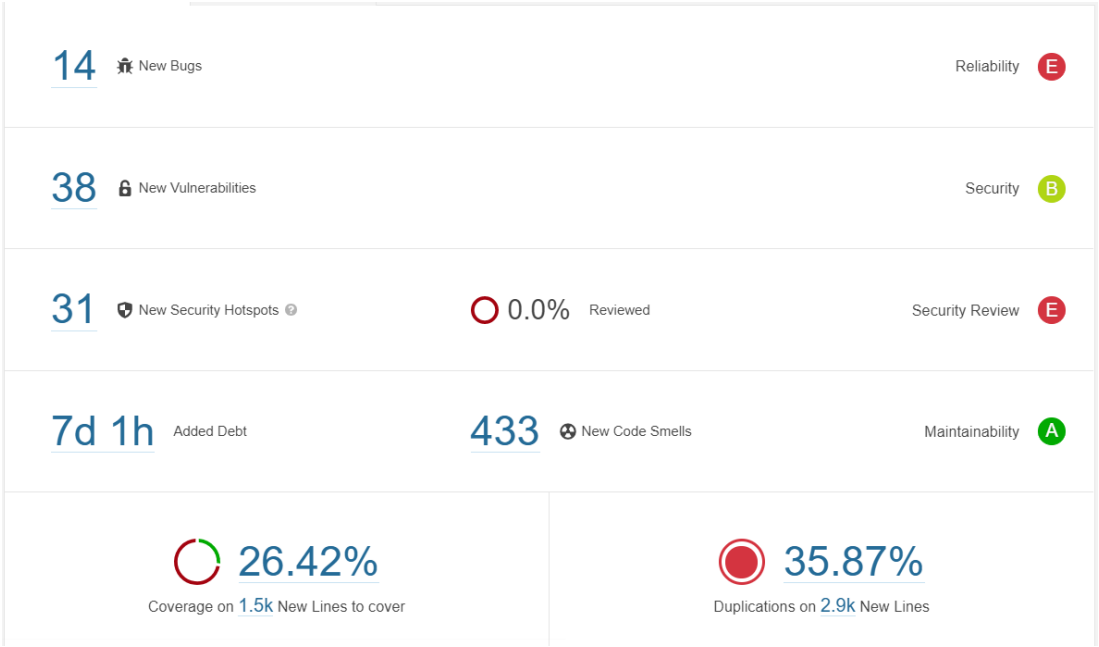
#Test Result = 20/21 = 대략 95.24%

4. Static Analysis

sonarqube 정적 분석 도구를 사용하여, jenkins에서 빌드한 파일을 정적 분석했다. findbugs, pmd, checkstyle을 플러그인으로 추가하여 도구 별로 rule set을 선정하여, 정적 분석을 수행했다.

전체 code smell이 많았기 때문에, bug와 major 이슈 중에서 유의미한 분석 결과만 추렸다.

A. Sonarway

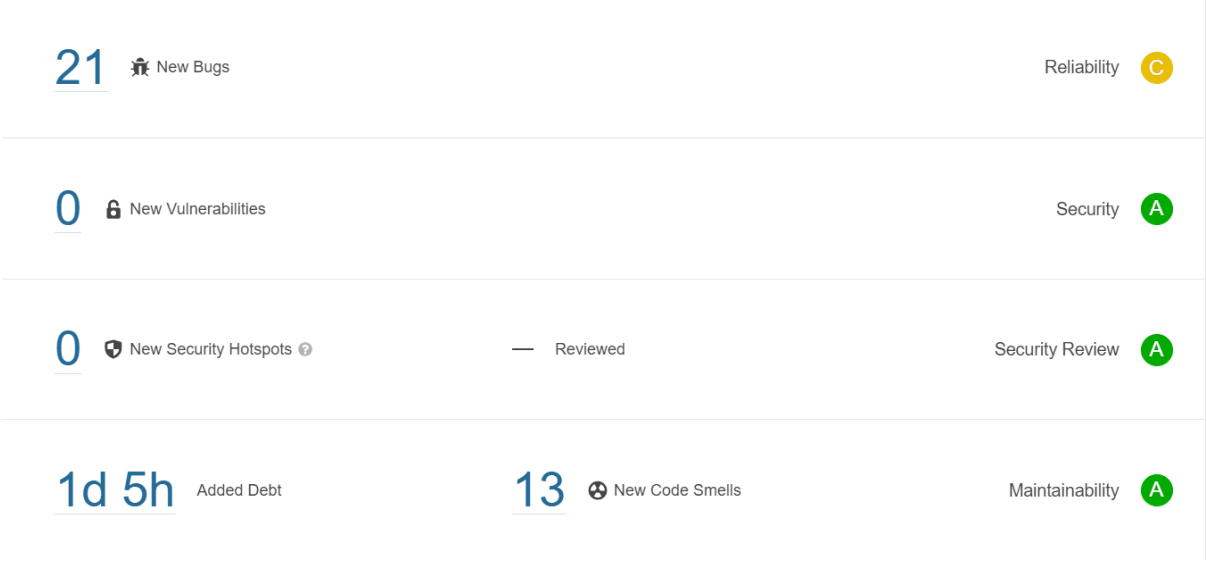


File	Line	Redmine issue#	Review
DVM1	75	130	Add a default case to this switch
	54	152	Call the method Thread.start() to execute the content of the run() method in a dedicated thread.
DVM2	71	133	Add a default case to this switch
	53	155	Call the method Thread.start() to execute the content of the run() method in a dedicated thread.
DVM3	71	134	Add a default case to this switch
	53	161	Call the method Thread.start() to execute the content of the run() method in a dedicated thread.
DVM4	71	135	Add a default case to this switch
	53	159	Call the method Thread.start() to execute the content of the run() method in a dedicated thread.
DVM5	71	137	Add a default case to this switch
	53		Call the method Thread.start() to execute the content of the run() method in a dedicated thread.
DVM6	71	139	Add a default case to this switch
	53	162	Call the method Thread.start() to execute the content of the run() method in a dedicated thread.
DVM7	71	131	Add a default case to this switch
	53	163	Call the method Thread.start() to execute the content of the run() method in a dedicated thread.
DVM8	71	136	Add a default case to this switch
	53	164	Call the method Thread.start() to execute the content of the run() method in a dedicated thread.
CodePayment	75	130	Add a nested comment explaining why ths method is empty, throw an UnsupportedOperationException or complete the implementation
Network	44	132	Use try-with-resources or close this "Socket" in a "finally" clause.
	73	138	Use try-with resources or close this "Socket" in a "finally" clause.
	7	141	Don't extend "Thread", since the "run" method is not overridden.
	126	146	Use try-with resources or close this "Socket" in a "finally" clause.
	18	156	Add a default case to this switch

MyFrame	15	144	Make "labelList" private or transient.
	168	147	Refactor this method to reduce its Cognitive Complexity from 64 to the 15 allowed.
StubTest	22	148	Add an end condition to this loop
	31	149	Add an end condition to this loop
	37	158	Add a default case to this switch
	66	160	Add a default case to this switch
OtherDVMs	4	157	Don't extend "Thread", since the "run" method is not overridden.

-> 전체 447개의 버그 및 code smell 중 29개의 유의미한 문제점 추려냄

B. FindBugs

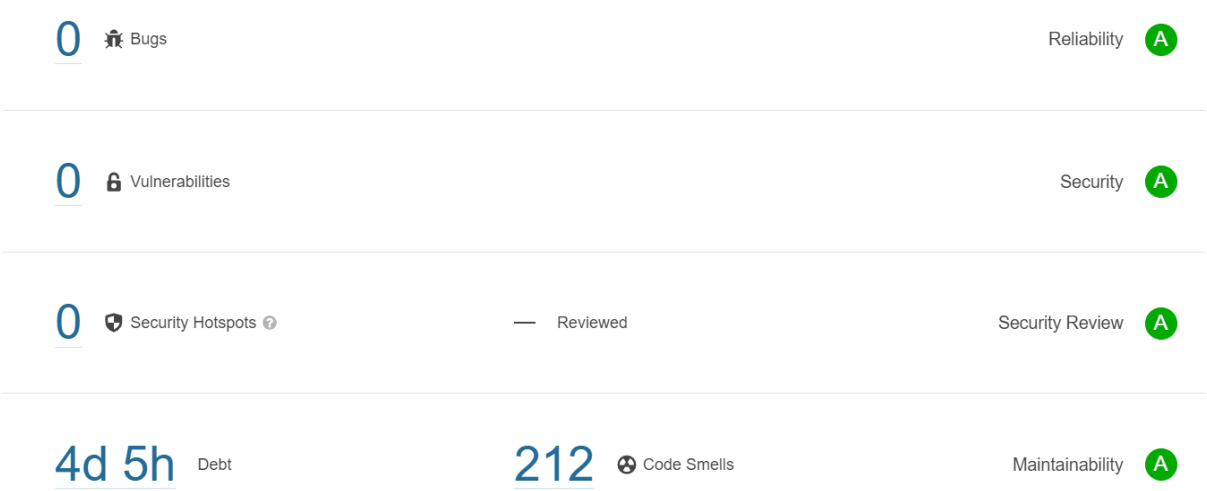


File	Line	Redmine issue#	Review
DVM1	75	185	Switch statement found in DVM1.run() where default case is missing
	54	184	DVM1.main(String[]) explicitly invokes run on a thread(did you mean to start it instead?)
	17	183	Unread field: DVM1.STUB_TEST_ID; should this field be static?
DVM2	71	188	Switch statement found in DVM2.run() where default case is missing
	53	187	DVM1.main(String[]) explicitly invokes run on a thread(did you mean to start it instead?)
	17	186	Unread field: DVM1.STUB_TEST_ID; should this field be static?
DVM3	71	191	Switch statement found in DVM2.run() where default case is

			missing
	53	190	DVM1.main(String[]) explicitly invokes run on a thread(did you mean to start it instead?)
	17	189	Unread field: DVM1.STUB_TEST_ID; should this field be static?
DVM4	71	194	Switch statement found in DVM2.run() where default case is missing
	53	193	DVM1.main(String[]) explicitly invokes run on a thread(did you mean to start it instead?)
	17	192	Unread field: DVM1.STUB_TEST_ID; should this field be static?
DVM5	71	197	Switch statement found in DVM2.run() where default case is missing
	53	196	DVM1.main(String[]) explicitly invokes run on a thread(did you mean to start it instead?)
	17	195	Unread field: DVM1.STUB_TEST_ID; should this field be static?
DVM6	71	200	Switch statement found in DVM2.run() where default case is missing
	53	199	DVM1.main(String[]) explicitly invokes run on a thread(did you mean to start it instead?)
	17	198	Unread field: DVM1.STUB_TEST_ID; should this field be static?
DVM7	71	203	Switch statement found in DVM2.run() where default case is missing
	53	202	DVM1.main(String[]) explicitly invokes run on a thread(did you mean to start it instead?)
	17	201	Unread field: DVM1.STUB_TEST_ID; should this field be static?
DVM8	71	206	Switch statement found in DVM2.run() where default case is missing
	53	205	DVM1.main(String[]) explicitly invokes run on a thread(did you mean to start it instead?)
	17	203	Unread field: DVM1.STUB_TEST_ID; should this field be static?
CardPaymeny	1	178	Unused field: CardPayment.card_info
	11	179	Unused field: CardPayment.isPrePayment
	30	180	.CardPayment.generateCode(Drink) uses the nextDouble method of Random to generate a random integer; using nextInt is more efficient
MyFrame	176	173	Dead store to aa in MyFrame$PadInput.actionPerformed (ActionEvent)
	201	174	Switch statement found in MyFrame$PadInput.actionPerformed(ActionEvent) where default case is missing

StubTest	11	175	Found reliance on default encoding in StubTest.main(String []):new java.util.Scanner(InputStream)
	37	176	Switch statement found in StubTest.test(int, int,int,String) where default case is missing.
Controllers	1	181	Unused field: Controller.card_info
	84	182	Integer is incompatible with expected argument type Code in Controller.insertCard(int, boolean)
Teset2	4	211	Dead store to m in Test2.main(String[])

C. PMD

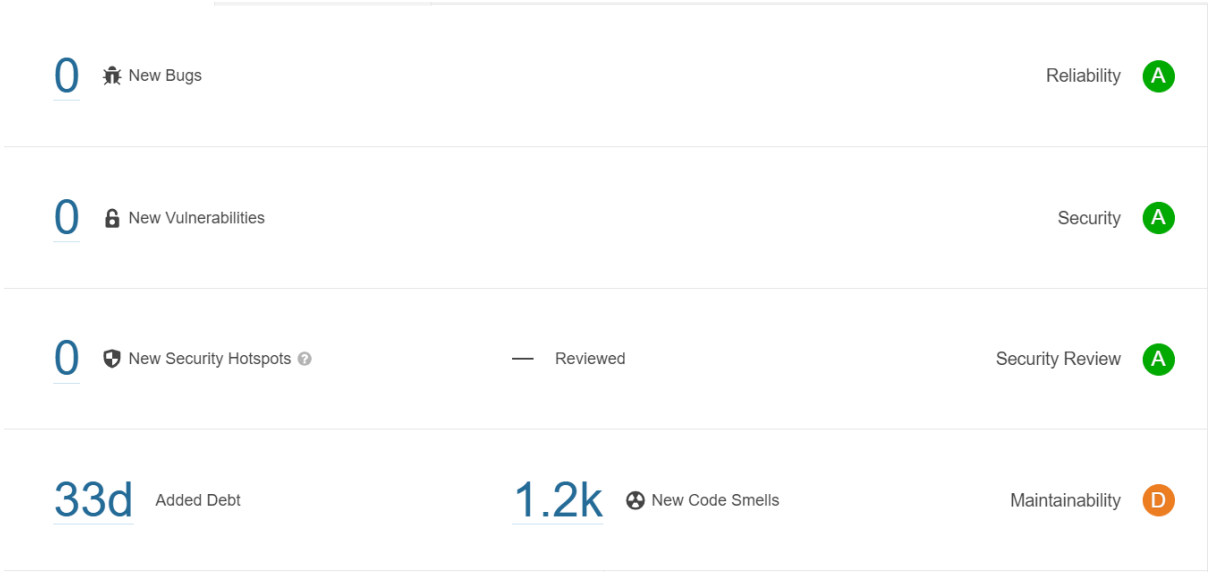


File	Line	Redmine issue#	Review
DVM	7	217	Avoid using implementation types like 'ArrayList'; use the interface instead.
	24	214	All classes, Interfaces, enums, and annotations must belong to a named package
StubTest	8	215	Avoid using final local variables, turn them into fields
	51	219	Switch statements should have a default label.
	53	220	Avoid catching generic exception such as NullPointerException, Exception in try-catch block
	80	223	Switch statements should have a default label.
	82	224	Avoid catching generic exceptions such as NullPointerException, RuntimeException, Exception in try-

			catch block
	86	218	The method 'test[int,int,int,String]' has a cyclomatic complexity of 17.
	87	212	Each class should declare at least one constructor
Card	2	216	Private field 'card_num' could be made final; it is only initialized in the declaration or constructor.
	19	213	All classes, Interfaces, enums and annotations must belong to a named package
Code	4	222	Private field 'drink' could be made final; it is only initialized in the declaration or constructor.
CodePayment	7	226	Avoid unnecessary constructors - the compiler will generate these for you
	7	229	Document empty constructor.
	23	230	Consider simply returning the value vs storing it in local variable 'drink_info'
Message	55	227	Classes implementing Serializable should set a serialVersionUID.
	55	228	Each class should declare at least one constructor.
Drink	6	221	Private field 'name' could be made final; It is only initialized in the declaration or constructor.
	9	225	Private field 'imgURL' could be made final; it is only initialized in the declaration or constructor.

-> 전체 212개의 버그 및 code smell 중 19개의 유의미한 문제점 추려냄

D. CheckStyle



- 중복해서 발생한 rule이 많아서 rule별로 묶어 표로 만들었다.

Rule	Review	Class-Line	Redmine issue #
Avoid Star Import	Using the '.*' form of import should be avoided - javax.swing.*.	MyFrame.java - L1 MyFrame.java - L2	231
Nested If Depth	Nested if-else depth is 2 (max allowed is 1)	MyFrame.java - L177	232
Visibility Modifier	Variable must be private and have accessor methods.	CardPayment.java - L9 Controller.java - L4, L5, L6, L7, L8, L9, L10, L11 DVM1.java - L13, L14, L15, L16 DVM2.java - L13, L14, L15, L16 DVM3.java - L13, L14, L15, L16 DVM4.java - L13, L14, L15, L16 DVM5.java - L13, L14, L15, L16 DVM6.java - L13, L14, L15, L16 DVM7.java - L13, L14, L15, L16 DVM8.java - L13, L14, L15, L16 MyFrame.java - L9, L10, L11, L12, L13, L14, L15, , L18, L19, L20, L21, L22, L25, L26, L29 Network.java - L8 OtherDVMs.java - L5, L6	233
Unused Imports	Unsued Imports (java.util.Arrays, java.io.IOException, java.util.ArrayList,java.util.List, java.net.ServerSocket,java.util.Arrays)	CardPayment.java - L2 DVM1.java - L1 DVM2.java - L1 DVM3.java - L1 DVM4.java - L1 DVM5.java - L1 DVM6.java - L1 DVM7.java - L1 DVM8.java - L1 Drink.java - L1, L2 Network.java - L3 OtherDVMs.java - L2	234
Iligal Catch	Catching 'Exception' is not allowed.	DVM1.java - L138 DVM2.java - L84, L134, L153, L179 DVM3.java - L84, L134, L153, L179	236

		DVM4.java - L84, L134, L153, L179 DVM5.java - L84, L134, L153, L179 DVM6.java - L84, L134, L153, L179 DVM7.java - L84, L134, L153, L179 DVM8.java - L84, L134, L153, L179 Network.java - L60, L91, L115, L141 StubTest.java - L53, L82	

-> 전체 1.2k개의 code smell 중 88개의 유의미한 문제점 추려냄

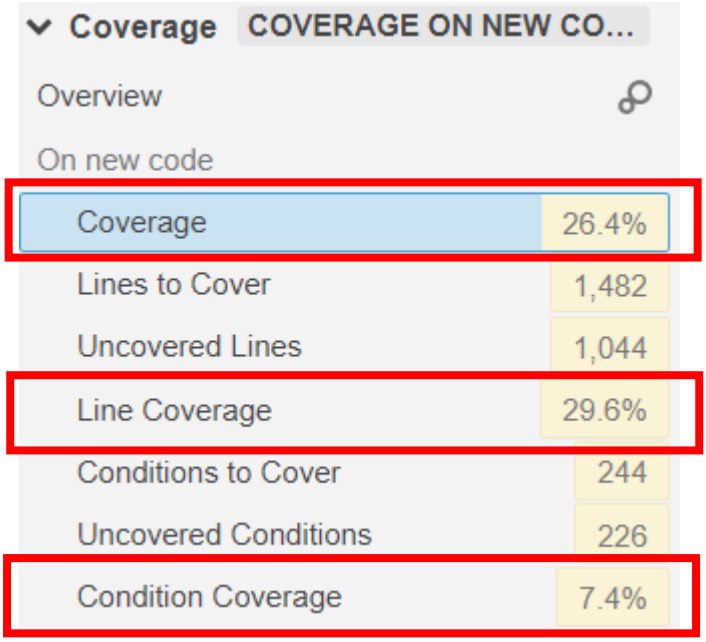
E. general review

전체적으로 if 문 중첩도가 높아 cyclometic cycle이 높게 나왔고, 쓰이지 않는 코드들이 많이 남아 있었다. 또한 switch문에 default문이 없는 경우가 많았다.

5. Coverage

Unit Test Case가 부족해서, 전체적으로 coverage가 매우 낮다.

- 전체 코드에 대한 coverage = 26.4%
- Line에 대한 coverage = 29.6%
- Condition coverage = 7.4%



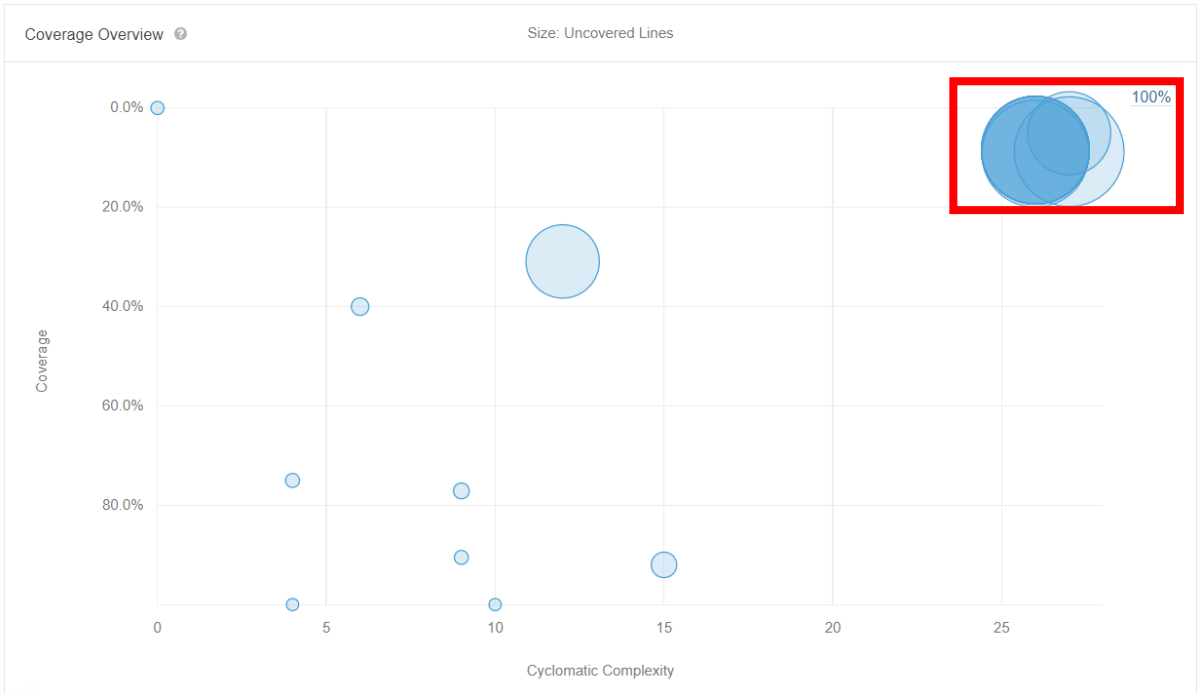
- 커버리지 제외 파일

GUI 및 test 파일은 유의미한 test 결과가 나오지 않기 때문에, coverage 계산에서 제외시켰다.

A. 전체 Coverage

- coverage graph

복잡도가 높은 클래스는 coverage가 낮고, 상대적으로 복잡도가 낮은 클래스는 coverage가 높게 나왔다.



- 각 클래스 별 전체 coverage

coverage가 100%가 나온 클래스도 있었지만, coverage가 0%가 나온 클래스도 존재한다.

-> DVM1~8.java : DVMTest 클래스에 실제 DVM1~8의 함수를 실행하는 테스트 코드가 없어 커버리지가 적게 나왔다.

-> CodePayment.java : Getter/Setter에 해당하는 Unit Test Code가 없어서 커버리지가 적게 나왔다.

-> MsgType.java : Unit Test Code가 없어서 커버리지가 0이 나왔다. assert equal을 이용해 MsgType의 코드를 확인해야 한다.

-> Controller.java : Controller의 주요 기능들을 테스트하는 유닛 테스트 코드가 @Disable 처리되어 있어서 커버리지가 적게 나왔다.

-> Network.java : 유닛 테스트 코드가 @Disable 처리되어 있어 커버리지가 낮게 나왔다.

	Coverage on New Code	Uncovered Lines on New Code	Uncovered Conditions on New Code
Card.java	100%	0	0
CardPayment.java	77.1%	4	4
Code.java	75.0%	2	0
CodePayment.java	40.0%	6	0
Controller.java	5.1%	80	32
Drink.java	90.5%	2	0
DVM.java	–	0	0
DVM1.java	8.8%	110	24
DVM2.java	9.2%	107	22
DVM3.java	8.5%	108	22
DVM4.java	8.5%	108	22
DVM5.java	8.5%	108	22
DVM6.java	8.5%	108	22
DVM7.java	8.5%	108	22
DVM8.java	8.5%	108	22
Message.java	100%	0	0
MsgType.java	0.0%	1	0
MyFrame.java	–	0	0
Network.java	30.9%	69	7
OtherDVMs.java	92.0%	15	5
StubTest.java	–	0	0
Test2.java	–	0	0

B. Line Coverage

- 각 클래스 별 line coverage = 29.6%

	Line Coverage on New Code	Uncovered Lines on New Code
Card.java	100%	0
CardPayment.java	85.2%	4
Code.java	75.0%	2
CodePayment.java	40.0%	6
Controller.java	7.0%	80
Drink.java	90.5%	2
DVM.java	–	0
DVM1.java	10.6%	110
DVM2.java	10.8%	107
DVM3.java	10.0%	108
DVM4.java	10.0%	108
DVM5.java	10.0%	108
DVM6.java	10.0%	108
DVM7.java	10.0%	108
DVM8.java	10.0%	108
Message.java	100%	0
MsgType.java	0.0%	1
MyFrame.java	–	0
Network.java	29.6%	69
OtherDVMs.java	93.7%	15
StubTest.java	–	0
Test2.java	–	0

C. Condition Coverage

- 각 클래스 별 condition coverage = 7.4%

Card.java	–	0
CardPayment.java	50.0%	4
Code.java	–	0
CodePayment.java	–	0
Controller.java	0.0%	32
Drink.java	–	0

DVM.java	–	0
DVM1.java	0.0%	24
DVM2.java	0.0%	22
DVM3.java	0.0%	22
DVM4.java	0.0%	22
DVM5.java	0.0%	22
DVM6.java	0.0%	22
DVM7.java	0.0%	22
DVM8.java	0.0%	22
Message.java	–	0
MsgType.java	–	0
MyFrame.java	–	0
Network.java	41.7%	7
OtherDVMs.java	64.3%	5
StubTest.java	–	0
Test2.java	–	0

D. Overview

1) 부실한 Unit Test

각 클래스 내의 함수에 대한 Unit Test Case가 없을 뿐만 아니라, 특정 클래스에 대한 Unit Test Case가 아예 없는 경우도 있었다.
따라서 전체적으로 Unit Test Case가 부족해서 Coverage가 낮게 나왔다.
또한 junit test에서 다음 테스트들이 fail 나 빌드가 되지 않아 @Disable 처리했다.
따라서 코드를 커버할 수 있는 테스트 수가 적어져 coverage가 낮게 나왔다.

```
ControllerTest > selectCurrentDrink() FAILED
    org.opentest4j.AssertionFailedError at ControllerTest.java:35

ControllerTest > selectOtherDrink() FAILED
    org.opentest4j.AssertionFailedError at ControllerTest.java:45

ControllerTest > startService() FAILED
    org.opentest4j.AssertionFailedError at ControllerTest.java:170

OtherDVMsTest > checkOtherDVMsStockTest() FAILED
    org.opentest4j.AssertionFailedError at OtherDVMsTest.java:39

OtherDVMsTest > showAccessibleDVMsLocationTest() FAILED
    org.opentest4j.AssertionFailedError at OtherDVMsTest.java:66

NetworkTest > handleRequestMessageTest() FAILED
    org.opentest4j.AssertionFailedError at NetworkTest.java:27

50 tests completed, 6 failed
```

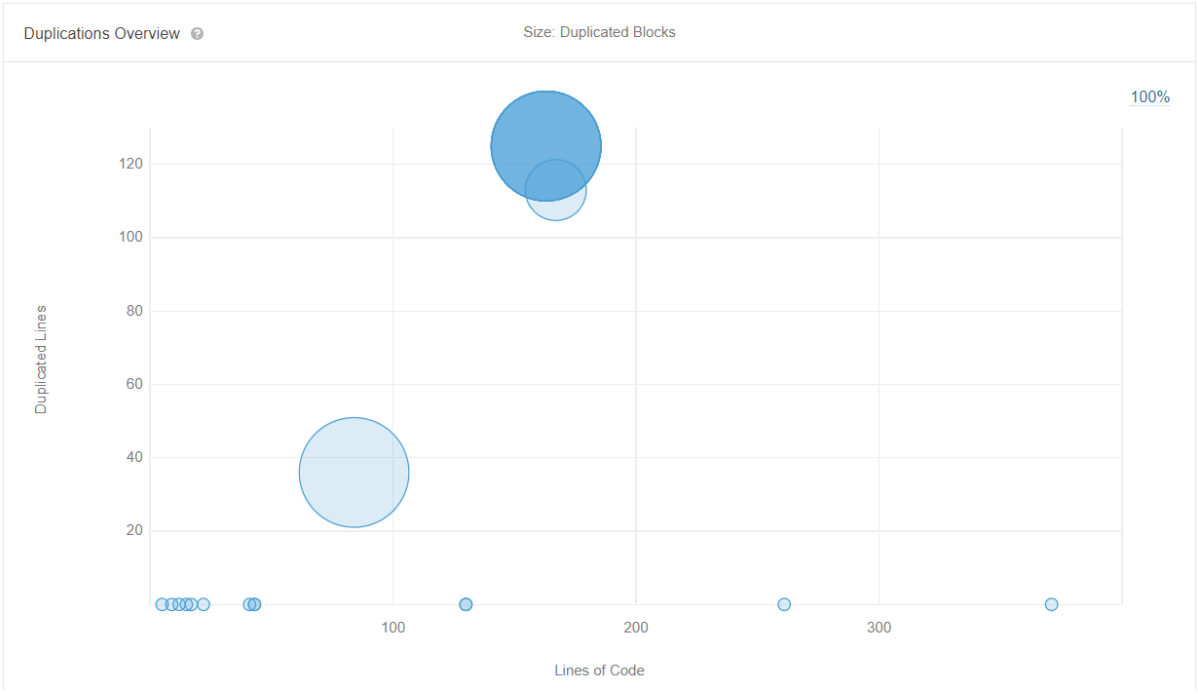
2) 중복 코드 발생

sonarqube의 다음 duplication을 참고하면, DVM1~8 클래스들은 약 70% 내용이 중복된다. 실제 코드를 확인해본 결과, 음료수 세팅 내용을 제외한 나머지 코드 내용이 동일한 것을 확인할 수 있었다. 이러한 방식으로 클래스를 중복 생성하기 보다는 DVM 클래스 하나에 대해 8개의 객체를 생성하는 것이 효율적이다.

- duplication이 높게 나온 클래스

DVM.java	0.0%	0
DVM1.java	60.4%	113
DVM2.java	68.3%	125
DVM3.java	68.3%	125
DVM4.java	68.3%	125
DVM5.java	68.3%	125
DVM6.java	68.3%	125
DVM7.java	68.3%	125
DVM8.java	68.3%	125

- 전체 코드의 라인 수 대비 중복 코드의 라인 수



6. 2nd Cycle Review 총평

spec리뷰와 system testing에서 알 수 있듯이 기능적인 측면은 잘 구현이 되었지만, network부분이 아직 미흡하다.
그리고 코드 내부적으로는 중복 코드가 많이 나와서, 객체 지향 언어인 java의 장점을 살리지 못했다.
또한 coverage가 낮게 나왔으므로 unit test code의 보강 및 수정이 필요하다.