REPORT

과 목:	프로젝트관리론
담 당 교 수:	정원석
제 출 일 자:	2023/12/14
전 공:	컴퓨터소프트웨어
학 번:	201904034
이 름:	최선재

목 차

Ⅰ. 요구사항분석

재고 관리 유스케이스명: 신제품 입고, 고객 반품 입고, 업체 반품 출고, 상품 출고, 현황 조회, 납품 업체 주문, 현황 반영, 현황 등록 엑터명: 입출고 담당자, 현황 관리 담당자, 쇼핑몰 시스템, 배송 직원, 납품 업체 유스케이스 개요: 입출고 담당자는 신제품이 입고되면 제품 상태를 확인하여 입고 또는 반품시킨다. 사전 조건: 현황 관리 담당자가 납품 업체에 주문한 제품이다.

이벤트 흐름

- 정상 흐름

- 1. 납품 업체에 제품 입고를 요구한다.

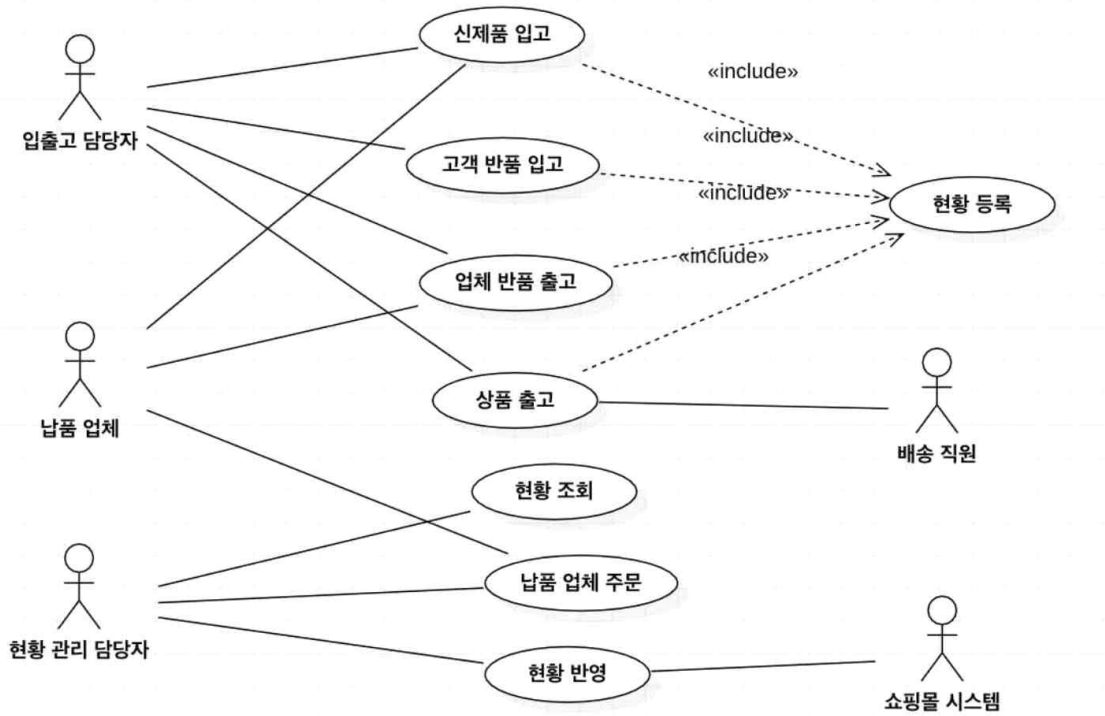
- 2. 입출고 담당자는 주문한 제품이 맞는지 확인한다.

- 3. 제품 상태를 파악한다.

- 4. 제품을 입고하고 제품 목록을 현황 관리 담당자에게 알린다.

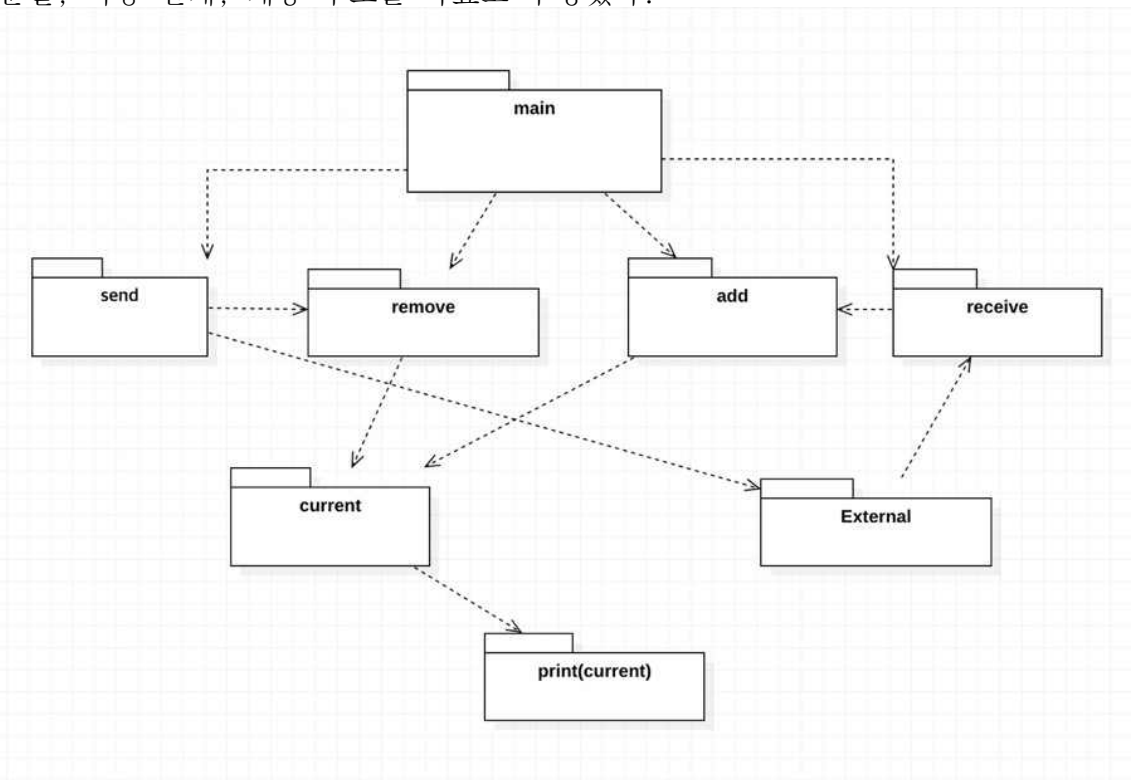
- 선택 흐름

- 1. 제품에 하자가 발생하면 현황 관리 담당자에게 하자를 알리고 반품한다.



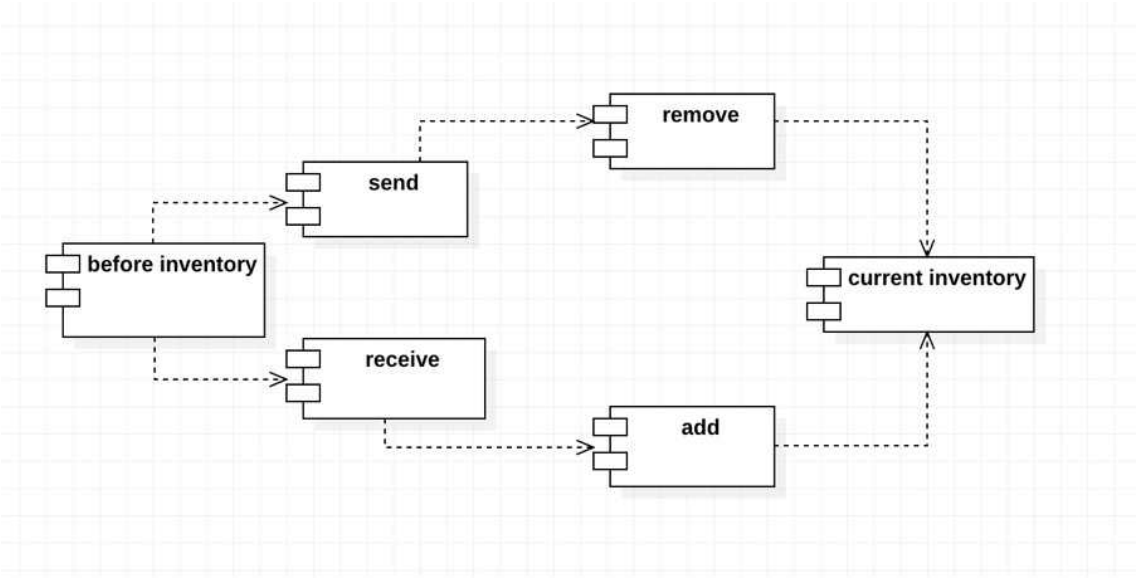
Ⅱ. 기본설계

기본설계의 첫 번째인 아키텍쳐 설계를 하겠다. 아키텍쳐 설계를 크게 2가지 관점으로 각 각 다이어그램을 통해 설계를 해보았다.
먼저 모듈 관점의 아키텍쳐 설계를 다이어그램을 통해 보여주겠다. 모듈관점은 일정 책임을 구현한 코드 단위인 모듈과 그 관계로 소프트웨어 구조를 설명한다. 분할, 사용 관계, 계층 구조를 목표로 구성했다.



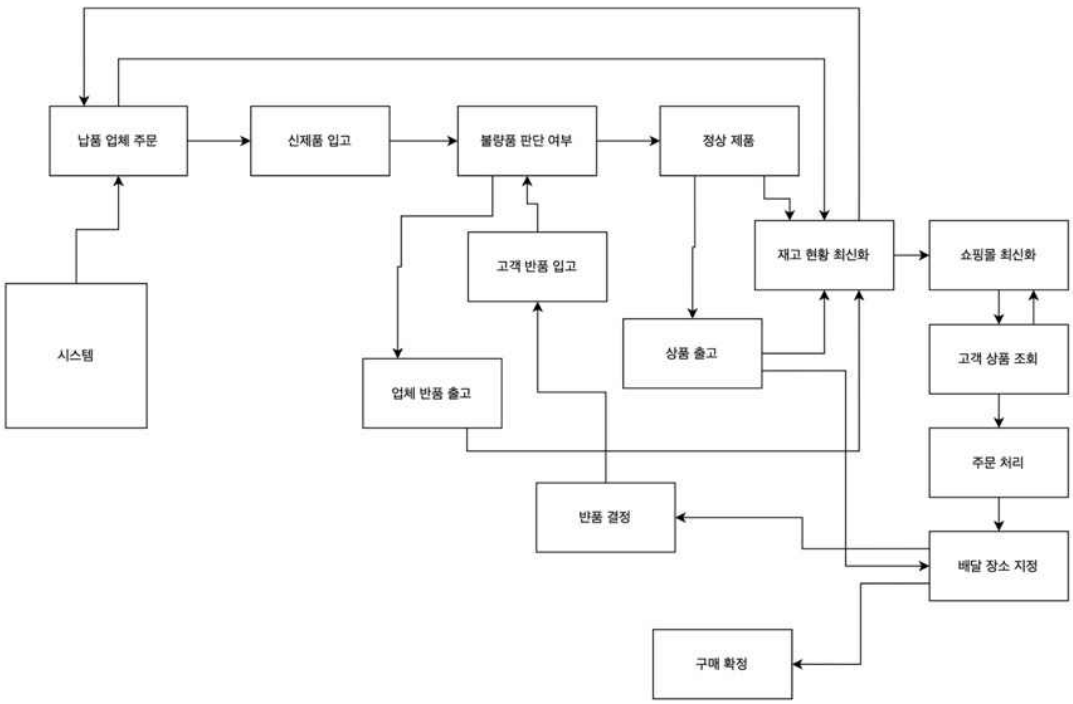
신제품 입고와 고객 반품 입고를 외부(External)에서 재고를 받았으면(receive) 재고에서 물품 수량만큼 더해(add) 현황 등록(current)를 한다. 반대로 업체 반품 출고, 상품 출고도 외부(External)로 보고 재품이 출고가 되었으면(send) 수량만큼 빼서(remove) 현황 등록(current)을 한다. 그 후 현황 반영,사용자 볼 수 있게 출력한다.(print (current))

두 번째로 컴포넌트 측면의 아키텍쳐 설계를 다이어그램을 통해 보여주겠다. 실행될 때 동작하는 요소와 상호작용으로 구조를 설명한다. 다시 말해서 더 추상적이라 표현할 수 있다. 컴포넌트 관점은 클라이언트 서버, 이벤트 중심 스타일로 두고 설계해보았다. 이벤트 중심이여서 어떤 식으로 작동하는지 간편하게 볼 수 있다.



이벤트(send, receive)가 발생하기 전 재고 현황(before inventory)이 이벤트를 만나 각각 add와 remove를 하여 새로운 재고(current inventory)로 저장하는 흐름이다.

다음으로 프로젝트를 서브 시스템으로 분할하여 아키텍쳐 설계를 마치겠다.

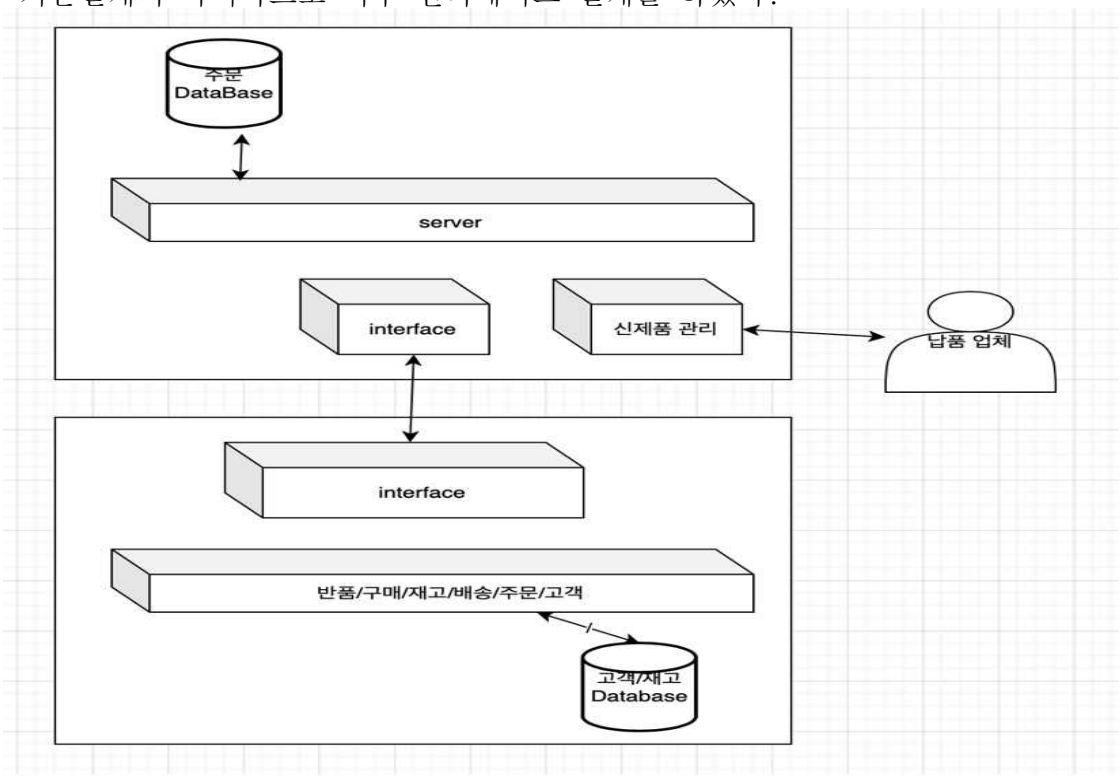


전체적인 시스템에서 먼저 납품 업체 주문을 할 때 재고 현황을 보고 신제품을 주문 후 입고를 한다. 입고 후 불량품 판단 여부를 확인한 뒤 정상 제품이면 해당 제품에 대한 재고가 증가하였으므로 재고 현황을 최신화한다. 불량품이면 업체 반품 출고를 진행하고 재고가 감소되었으므로 재고 현황을 시킨다. 재고 현황 최신화는 쇼핑몰 업데이트를 하여 고객이 상품을 조회할 수 있게 한다. 고객은 해당 상품을 주문하고 배달장소를 지정하면 상품 출고를 통해 상품이 발송되어 진다. 상품이 출고를 했으므로 재고를 최신화 해준다. 만약 고객이 반품을 결정을

했으면 고객 반품 제품 입고를 해서 다시 불량품 판단 여부를 하여 정상적인 흐름으로 진행한다.

다음으로 Ui설계를 하겠다.
관리자 UI: 납품 업체 주문, 불량품 재고 확인, 불량품 출고, 현황등록(재고 추가,재고 감소), 쇼핑몰 최신화,상품 출고, 반품 제품 목록 확인
고객 UI:물품 조회, 주문, 배달 장소 지정,환불 신청,주문 내역 확인, 배송 확인
배송 직원 UI:배송 내역확인, 고객 정보 확인, 배송, 반품 제품 회수,배송 현황 전송
쇼핑몰 UI: 상품 등록, 재고 수정, 제품 삭제

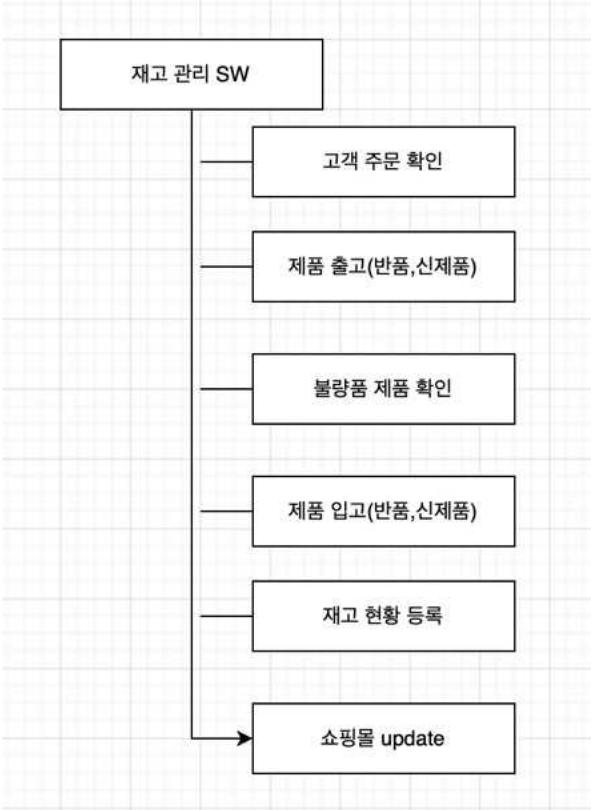
기본설계의 마지막으로 외부 인터페이스 설계를 하겠다.



관리자가 납품 업체에 신제품에 대한 필요 요청을 납품 업체에 함. 그 후 납품 업체는 요청을 받고 필요 재고만큼 제품을 보냄. 그리고 불량품 제품 출고에 대한 요청을 납품 업체에 함. 요청을 받았으면 입고 대기함.

Ⅲ. 상세설계

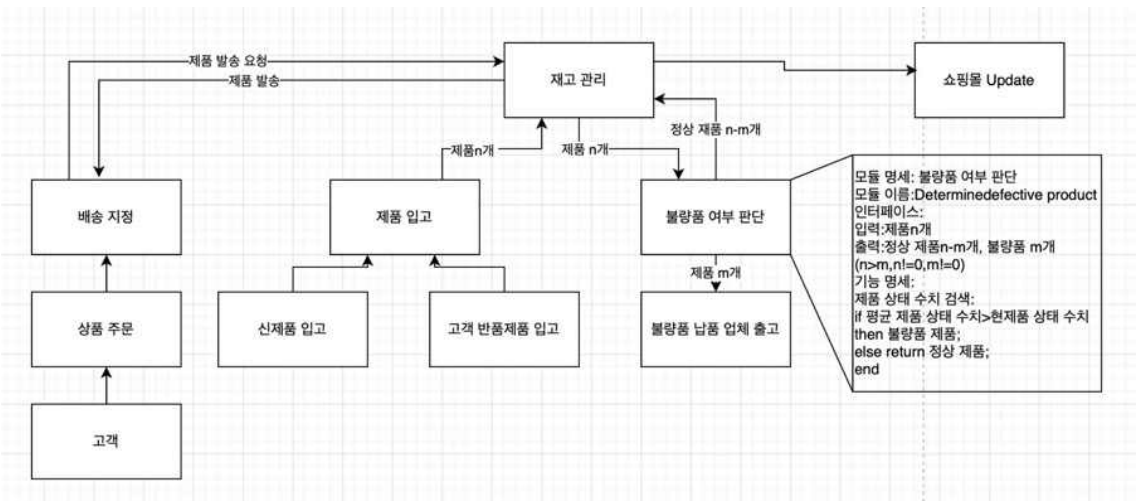
상세 설계의 모듈 분할 설계다.



최대한 모듈분할의 장점인 높은 응집력과 낮은 결합으로 독립적인 형태로 처리할 수 있도록 구성하였다.

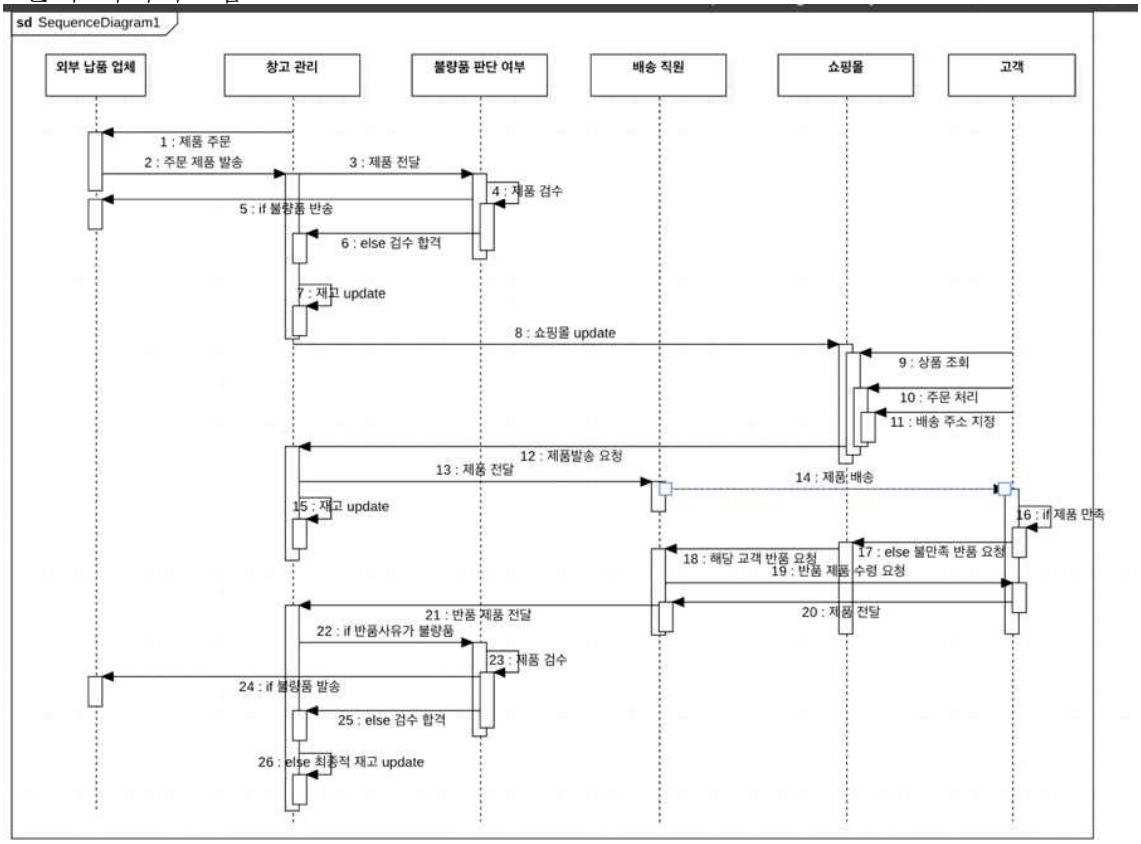
모듈 명세서

2번째로 처리 로직 설계다.

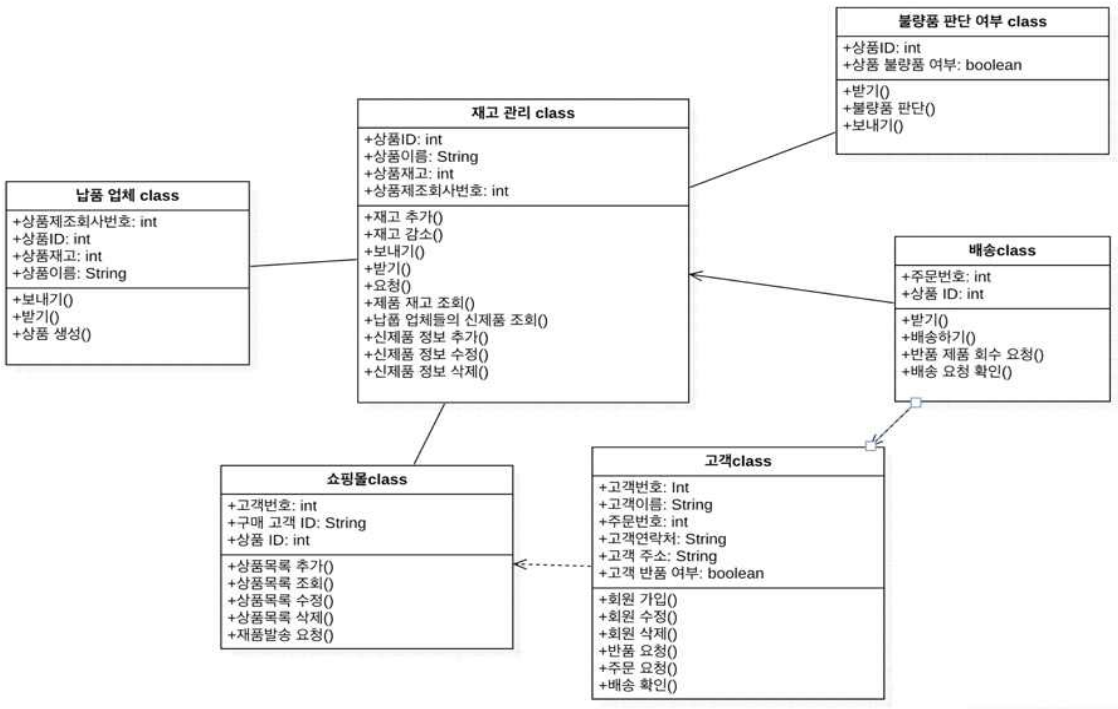


모듈 안의 처리 로직 설계를 하여 독립적인 목적을 완수해야 할 최소 단위의 업무 프로세스를 나타내 보았다.

순차 다이어그램:



클래스 다이어그램:



Ⅳ. 알고리즘

관리자는 처음에 관리자로 회원가입 후 로그인함.→관리자는 제품에 대한 재고를 조회, 납품 업체들의 신제품 조회를 함→수량이 부족하거나 신제품 입고를 원하면 외부 납품 업체에 주문을 요청→외부 납품 업체는 요청을 받고 제품을 보냄→창고가 해당 제품을 받고 바로 불량품 판단에 제품을 전달→불량품 판단에서 해당 제품 상태 수치가 평균보다 낮으면 불량품이라 판단하고 외부 납품 업체에 반송, 아니면 창고에 보냄→ 관리자는 신제품에 대한 정보를 추가→창고는 정상제품이 들어왔으므로 해당제품에 대한 재고를 추가함→쇼핑몰에 update함.

사용자는 처음에 회원가입후 로그인함→쇼핑몰의 제품을 조회→주문하기→배달 주소 지정→쇼핑몰은 관리자에게 재품 발송 요청→관리자는 발송 요청을 받고 제품을 배송 직원에게 전달 후 재품에 대한 재고 감소→배송 직원은 고객 주소로 재품을 발송→사용자는 배송 확인 후 제품에 만족하면 종료 불만족이면 쇼핑몰에 반품 요청→쇼핑몰은 배송 직원에게 고객 반품 요청→배송 직원은 고객에게 수령 요청을 한 후 해당 제품을 받음→제품 창고로 전달→반품 사유가 불량품이면 불량품 판단을 보내고 아니면 재고 추가.

배송 직원은 처음에 회원가입후 로그인함→배송 요청을 확인 후 재품을 받음→재품을 배송.

Ⅴ. 테스트

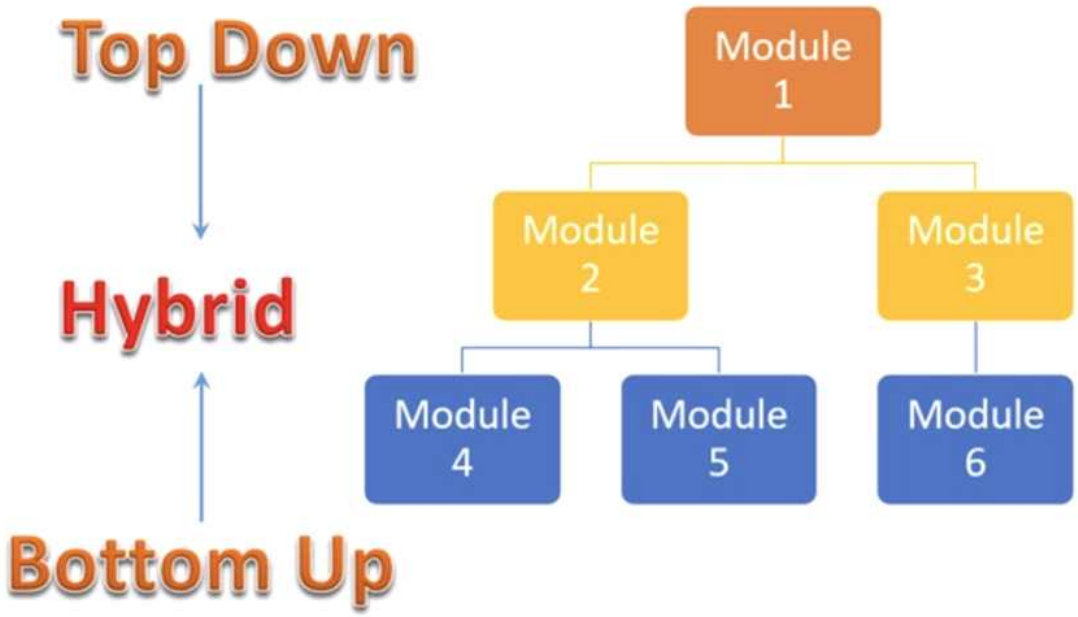
단위 테스트 단위의 크기가 작을수록 단위의 복잡성이 낮아진다. 따라서, 단위 테스트를 활용하여 동작을 표현하기 더 쉬워진다. 즉, 테스트 대상 단위의 크기를 작게 설정해서 단위 테스트를 최대한 간단하고 디버깅하기 쉽게 작성해야 한다. 개발자 관점에서 테스트를 하기 때문에 단위 테스트는 소프트웨어 내부 코드에 관련한 지식을 반드시 알고 있어야 하는 화이트박스 테스트를 알고 있어야 한다. 단위 테스트는 TDD와 함께 할 때 특히 더 강력해진다. TDD(Test-Driven Development,TDD)란 테스트 주도 개발이다. 반복 테스트를 이용한 소프트웨어 방법론으로 작은 단위의 테스트 케이스를 작성하고 이를통과하는 코드를 추가하는 단계를 반복하여 구현한다. 짧은 개발 주기의 반복에 의존하는 개발 프로세스이며, 애자일 방법론 중 하나인 eXtream Programming(XP)의 'Test-First' 개념에 기반을 둔 단순한 설계를 중요시한다.

먼저 단위테스트를 해보겠다. 모듈에 대한 테스트를 의미한다. 코드가 효율적으로 작성되었는지, 프로젝트 내에 합의된 코딩 표준을 준수하고 있는지도 검증한다.

- 클래스에 있는 입력 변수의 개수가 정확한지 확인한다.
- 입력 변수의 타입이 정확한지 확인한다.
- 변수 이름이 부정확하게 타이핑되었거나 컴파일러에 의해 잘리지 않는지 확인한다.
- 변수의 스코프를 확인하여 참조가 잘 되어있는지 확인한다.
- 함수를 호출할 때 인자개수가 정확한지 확인한다.
- 테스팅을 하면서 예를 들어 개발하면서 stage 3를 테스트 할 때, 항상 stage 1, stage 2를 클리어한 뒤 테스트를 해야한다.테스트 비용이 증가한다. 이럴 때에 어떻게 하면 비용을 낮출 수 있을까를 고민한다.(TDD)

다음으로 통합 테스트를 한다. 통합 테스트 단계에서는, 각각의 모듈들을 통합하여, 통합된 컴포넌트간의 인터페이스와 상호작용 상의 오류를 발견하는 작업을 수행한다. 통합 테스트를 하는 이유는 모듈 개발 시 고객의 요구 사항이 변경될 가능성이 크다. 이러한 새로운 요구 사항은 단위 테스트가 불가능할 수

있으므로 시스템 통합 테스트가 필요하다. 또, 데이터베이스와 소프트웨어 모듈의 인터페이스에 오류가 있을 수 있다.
 통합 테스트에선 샌드위치 테스트를 하겠다. 샌드위치 테스트 최상위 모듈은 하위 모듈로 테스트하고, 동시에 하위 모듈은 최상위 모듈과 통합하여 하나의 시스템으로 테스트하는 전략이다.즉, 하향식과 상향식의 장점을 이용하는 방식이다. 스텁과 드라이버를 모두 사용한다.
 병렬 테스트가 가능하고 시간이 절약이 가능하다. 팀원이 있으면 좋은 기법이라 생각한다.



 그 다음으론 회귀 테스트를 진행한다. 이전의 실행 테스트를 재실행하여 이전에 고쳐졌던 오류가 다시 재현되는지 검사하는 방법으로 많이 사용된다. 따라서 변경된 소프트웨어 구성 요소에 대해 집중적으로 회귀 테스트를 수행해야 한다. 먼저, 현재 테스팅을 하는 실제상황에 따라 회귀 테스트를 나눠보겠다. 시간이 많이 남은 시점에서는 Retest All기법을 사용한다. 기존에 가지고 있던 모든 회귀 테스트 및 데이터들을 다시 이용하여 테스트를 진행한다. 테스트 커버리지는 향상될 수 있겠으나 테스트에 시간이 많이 걸린다는 단점이있다. 시간이 별로 없는시점에서는 Selective기법을 사용하여 신규 기능을 검토하여 영향이 있을 것 같은 기존 기능들을 선택적으로 추려내어 테스트를 진행한다. 테스트 수행 범위가 최소화되어 시간과 코스트를 절약할 수 있으나, 테스트 커버리지가 부족하여 프로젝트에 결함이 존재할 확률이 높아진다.

 그 다음으론 시스템 테스트를 진행한다. 소프트웨어와 다른 시스템 요소(하드웨어, 등)들과 통합하여야 하며, 모든 요소들이 적절히 조화를 이루어 시스템의 기능을 만족하는지 확인하는 시스템 테스트를 수행해야 한다. 또한, 모듈이 통합된 후 시스템의 기능 및 비기능 요구 사항을 모두 평가하기 위해 시스템 테스트를 수행한다. 최종적으로 시스템의 성능이나 기능에 제한이 없는지 확인하는 단계이다.

복구 테스트
보안 테스트
스트레스 테스트
성능 테스트기능
테스트사용성 테스트
 마지막으로 인수 테스트를 하며 테스트를 마치도록하겠다. 개발된 소프트웨어가 고객의 요구사항을 만족하는지 조사하는 시험을 확인 테스트라고 한다. 확인 테스

트는 개발자가 하는 반면 인수 테스트가 사용자 입장에서 수행하는 것이다. 인수 테스트 중에선 알파 테스트와 베타 테스트가 있다. 알파 테스트는 사용자를 개발자 환경으로 초대해서 진행하는 테스트이다. 내가 사용자한테 사용하는 모습을 보여주면서 즉각 피드백을 얻는 테스트이다. 그 즉시 피드백을 하여 사용자의 요구사항을 듣고 판단하는 테스트가 된다. 베타 테스트는 다수의 사용자들이 각자의 환경에서 진행하는 테스트이다. 주변 사람들한테 출시 전 배타 테스트를 줘 다양한 사람들로부터 피드백을 받도록 한다.

VI. 형상관리

코드가 변경될 때마다 기록을 해둬서 다른 사람이 봤을 때도 어느 시점으로 어디가 바꼈는지 한눈에 볼 수 있게 깃에 저장해 놓는다.