POS 시스템 [스프링프레임워크]

1. 데이터베이스, 객체
데이터베이스는 새로운 데이터베이스 posdb를 생성하여 사용하였습니다. 사용된 데이터 테이블은 계정들의 정보를 가질 member 테이블, 제품들의 정보를 가질 product 테이블, 마지막으로 판매 기록을 저장할 recode 테이블을 사용하였습니다. 우선 member테이블은 계정 아이디, 비밀번호, 이름, 직책 등 여러 개인정보로 구성되어 있습니다. 직책을 추가한 이유가 권한 별로 접근할 수 있는 기능들을 다르게 하기 위함입니다. 총 3가지로 구성되어 있습니다.
admin, manager, parttimer로 구성되어 있습니다. admin은 모든 권한과 삭제가 불가능한 계정입니다. manager는 모든 기능들을 사용할 수 있고, parttimer는 계정 목록, 계정 삭제, 제품 수정 등 같은 메뉴를 사용할 수 없게 설정해 두었습니다. 다음은 product 테이블은 제품 아이디, 제조사, 가격, 입고량, 최근 입고 일자 등으로 구성되어 있습니다. 최근 입고 일자를 추가한 이유는 실생활에서 최근 입고 일부터 얼마만큼의 물건이 판매 되었는지 가늠하기 위함입니다. 다음은 recode 기록 테이블입니다. 기록번호, 제품 아이디, 제품 정보, 판매 날짜 그리고 recode_type 이라는 열을 추가하여 실제로 판매되었는지 환불 되었는지 구분하기 위해 추가하였습니다. 또 recode의 기본키는 자동으로 증가하는 정수값으로 설정해 두었습니다. 환불 목록도 확인하기 위해서 이 테이블은 프로젝트 내부에서 삭제 불가능 하도록 제작하였습니다.

다음은 프로그래밍에 사용되는 객체들을 설명하겠습니다. 요번 프로젝트에 사용된 객체들은 데이터베이스와 매칭하기 위한 member, product, recode객체, 그리고 프로젝트에 사용되는 cart, SumAmount, recode2 객체가 있습니다.
cart객체는 제품을 계산할 때에 현재 계산중인 제품들을 다루기 위한 객체입니다. 그리고 product에서 계산에 필요 없는 정보들을 제거하여 새로 만든 객체입니다. SumAmount객체는 판매량, 총액량 통계에 사용되는 객체입니다. 이 통계들은 recode의 판매시간, 판매순서 같은 필요 없는 멤버 변수들을 없애고 쉽게 통계를 구하기 위한 객체입니다.
recode2객체는 시간과 관련된 통계를 다루기 위해 만든 객체입니다. 기존 recode의 객체들의 변수의 개수는 같지만 모든 멤버변수들을 문자열로 만들었습니다. 이렇게 만든 이유가 시간순서대로 통계를 제작하려면 기간이 표시되어야 하는데 recode의 시간 변수 타입 LocalDataTime의 예시는 “2022-06-05 12:30:55” 이런 식으로 되어 있는데, 문자열로 변환하고 적당한 길이로 문자열을 자르면 연도, 월, 일로 나눌 수 있고, 그 자른 시간 시간테이블을 테이블 형태로 저장하기위해 문자열로 제작하였습니다. 또 정수형 문자열의 기본 값이 0이고, 문자열의 기본 값은 “”이기 때문에 시간테이블의 필요 없는 출력을 줄이고자 모든 멤버 변수를 문자열로 만들었습니다. 그리고 시간 타입을 LocalDataTime으로 설정한 또 다른 이유는 일주일 통계를 위해서 이기도 합니다. 이유는 LocalDataTime 클래스의 plusweek(int)라는 함수를 사용하면 7일 단위로 다음 주의 날짜를 알 수 있어서 사용하였습니다.

2. Java 코드
프로그래밍에 사용되는 객체들은 object패키지에 담아서 사용하였습니다. 객체, 기본으로 생성되어있는 java소스를 제외하면 총 6개의 java 코드로 되어 있습니다. Bean과 의존성 주입을 위한 BeanConfig, jsp 코드들을 매핑하기 위한 MainControllar, member, product, recode, cart객체들을 다루기 위한 Dao코드로 구성되어 있습니다.
프로그램 전체로 보면 메인, 판매, 통계, 재고, 계정 5개의 파트로 나눌 수 있는데 프로젝트를 제작하면서 5개의 컨트롤러로 만들려고 하였으나 나눠서 작성하게 되면 매핑을 못하는 경우가 생겨서 MainControllar에는 모든 프로젝트의 매핑 코드가 들어있습니다. 또 요번 프로젝트에서 가능하면 Post방식으로 사용하였고 한 페이지에 여러 개의 Post가 있는 경우도 있어서 코드의 순서, 조건같은 것을 잘 고려하여 코드를 작성하였습니다. member객체들을 다루기 위한 memberDao에서는 계정 목록 출력을 위한 모든 member 출력, 로그인을 위한 select 함수, 계정 생성 시에 아이디 중복을 막기 위한 중복방지, 아이디로 계정 삭제 등의 함수로 구성되어 있습니다. productDao는 제품 목록 출력을 위한 함수, 제품 재 입고, 새 제품 입고, 환불 시 재고 증가, 판매 시 재고 감소 같은 함수들로 구성되어 있습니다. CartDao는 제품 판매 시 여러 종류의 제품들을 판매를 다루기 위해 만들었고, 장바구니 추가, 비우기 같은 함수들로 구성되어 있습니다. recodeDao는 recode객체와 또 통계를 위한 SumAmount, recode2 객체들을 다루는 함수들로 구성 되어있습니다. SumAmount, recode2를 이용하는 기능들을 얼마 안 되서 같은 공간에 소스를 작성하였습니다. 기록들을 출력, 환불 되지 않는 기록들만 출력, 제품 판매 시 기록 추가, 총액, 수량 같은 통계를 위한 같은 제품아이디로 묶어서 판매량 순, 총액 순으로 데이터베이스를 출력하는 함수, 시간 순으로 정렬해서 데이터베이스를 출력하는 함수, 시간 단위의 통계를 위한 통계함수 특히, 연간, 월간, 일간 통계 기능을 LocalDataTime타입의 변수의 앞에서 자르는 길이를 조절하여 연간, 월간, 일간 정보를 얻어서 하나의 함수로 제작하였습니다.

MemberDao.java	
List<Member> selectAll ()	모든 멤버 데이터 리턴
List<Member> selectListAll (String)	String의 아이디를 가진 멤버 리스트를 출력, 하나 뿐이라서 Member를 반환형으로도 가능해지만 jsp 코드에 맞게 List 반환형
insert (...)	insert 쿼리문
update (...)	update 쿼리문
void deleteById(String)	계정 삭제 쿼리문

Member login (String, String)	로그인 쿼리문
boolean is_ID_overlap(String)	아이디 중복여부에 대한 쿼리문

ProductDao.java	
List<Product> selectAll ()	모든 멤버에 대한 데이터 리턴
Product selectByID	해당 제품아이디의 product 객체 리턴
List<Product> selectListByID	위의 함수를 jsp에서 다루기 쉽게 list 형식으로 변경
insert(...)	insert 쿼리문
public void updateById()	update 쿼리문
restock(...)	재입고 함수, 최근 입고 날짜 현재 시간으로 초기화
sell(...)	제품 판매에 사용되는 업데이트 쿼리문 함수
refund(...)	환불에 사용하는 업데이트 쿼리문 함수
deleteById(String)	제품폐기에 사용되는 delete 쿼리문 함수

RecodeDao.java	
List<Recode> selectAll ()	모든 멤버에 대한 데이터 리턴
List<Recode> selectListByID (int recodeID)	해당ID의 recode 정보 list형식으로 반환
Recode selectByID (int recodeID)	해당ID의 recode 정보 recode형식으로 반환
List<Recode> selectListByNormal ()	환불 기록을 제외한 모든 판매기록
insert()	제품 판매 시 기록을 하기위한 insert
change_refund()	제품 판매 기록을 환불 기록으로 변환
List<SumAmount> groupingbyamount()	수량 순 통계를 하기위해 같은 제품 아이디로 그룹으로 만들어서 필요한 정보들을 반환
List<SumAmount> groupingbyTotal()	총액 순 통계에 사용되는 총액 순으로 정렬된 데이터
List<Recode> sortDate()	날짜별 통계에 사용하기 위해 날짜 순으로 정렬된 데이터
Recode2 toR2(Recode)	Recode객체를 Recode2객체로 변경
List<Recode2> statWeek()	주간 통계 함수
List<Recode2> statDate(String date)	연간 월간 일간 통계 함수 매개변수에 "year" "month" "day" 입력

CartDao.java	
add(...)	장바구니 추가 함수
deleteAll()	판매완료 시 장바구니 비우기
getCartIndexById(String)	제품의 인덱스가 어디인지 찾는 함수, 제품의 개수를 추가 할 때 중복된 아이디가 있으면 해당 인덱스에서 개수 추가
getTotal()	총액을 계산하는 함수

3. jsp 코드

jsp코드는 메인, 계정, 재고, 판매, 통계로 총 5개의 파트로 구별되어서 제작하였습니다. 파트별로 로그인, 직책 세션을 가져 접근 권한을 제한하면서 기능별로 버튼들을 모아두는 메뉴 jsp를 각 파트별로 만들었습니다. 메인파트에서는 로그인, 기본 홈, 그리고 재고 파트와 판매 파트에서 동시에 사용되는 제품 리스트페이지로 구성되어 있습니다.

리스트를 출력하는 종류의 페이지 같은 경우에는 컨트롤러에서 뷰로 데이터를 전달하는 방식으로 구현하였습니다.

계정 파트는 계정 생성, 삭제, 수정, 계정리스트 출력 4가지의 기능들로 구성되어 있습니다. 계정 생성, 삭제, 리스트 출력 같은 경우에는 계정파트의 메뉴의 세션을 이용하여 기능 접근을 제한하였습니다. 계정 수정이랑, 생성에는 중복 방지를 위한 유효성검사를 추가하였고, 계정 삭제 같은 경우에는 계정 목록을 출력하고 삭제 진행할 경우 한 번 더 정보들을 확인하고 삭제시기키 위해서 컨트롤러와 뷰 사이의 데이터 전달을 중복 사용하였습니다.

재고 파트는 재고 확인, 제품입고, 재 입고, 폐기 기능을 가지고 있습니다. 제품 수정 같은 경우 수정 버튼을 누르면 제품의 원래 정보가 사라지는 것이 아니고 원래 정보들을 유지하기 위해 컨트롤러와 뷰의 데이터 전달을 또한 중복적으로 사용하였습니다. 판매 파트에서는 판매 기록, 환불, 제품 판매의 기능이 있습니다. 제품 판매의 기능 같은 경우 장바구니에 상품을 추가할 때 페이지가 넘어가지 않도록 하기위해 컨트롤러와 뷰의 데이터 전달을 중복하여 사용하였고, 결제 창으로 페이지가 전환될 때에 장바구니의 정보를 저장하기 위해서 CartDao.java를 생성하였습니다.

마지막은 통계 파트입니다. 통계파트에서는 연간 월간 일간 통계 같은 경우에는 날짜만을 보고 파악이 가능하지만, 주간 단위의 통계 부분에서 그렇지 않아서 시간이 꽤 소요되었습니다. 주간 통계는 날짜순으로 정렬된 데이터들이 필요하고, 처음 데이터의 날짜를 기준으로 잡아서, 다음 데이터가 해당 주에 해당되는지 확인 합니다. 해당 주에 해당되지 않으면, 그 주의 통계의 총합을 저장하고 다음 데이터가 해당하는 주가 될 때까지 7일씩 더해가면서 날짜를 비교하면서 같은 주의 데이터 끼리 묶었습니다. 또 통계파트에서는 특히 통계의 기준을 작은 메뉴로 만들어서 시각화 효과를 높였습니다.