[비디오 가게 관리 시스템]

소프트웨어 공학

<설계 명세서>

2018년 12월 07일

소　　속 :　충북대학교 소프트웨어학과

이　　름 :　구 경 민, 정 예 원

교　　수 :　이 종 연 교수님

Contents

1.서론

현재는 스마트폰 대중화로 온라인으로 상품이나 서비스를 주문하고, 오프라인에서 소비하는 형태로 자리 잡아가고 있다. 비디오 매장, 매출, 회원 관리, 대여 사업, 공급 업체 등을 직접 관리하지 않고 소프트웨어를 통해 관리할 수 있게 한다. 소비자역시 PC나 스마트폰으로 직접 소비, 대여를 할 수 있다.

1.1 개발 목표

이 시스템의 개발목표는 소형 비디오가게 관리를 위한 데이터베이스시스템의 개발을 목표로 한다. 그 세부적인 개발내용은 다음과 같다.

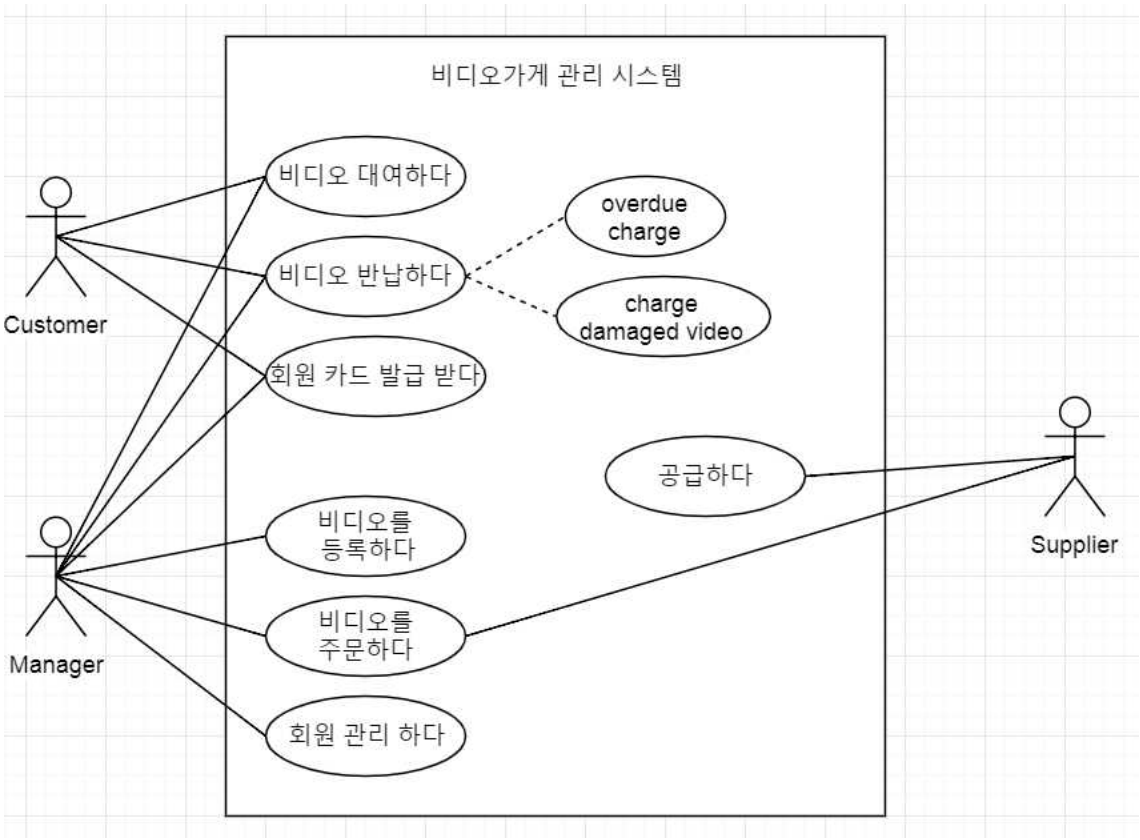
-소비자는 비디오 대여, 정보조회 등을 쉽게 할 수 있다.
-소비자는 회원 카드를 발급받고 자유롭게 서비스를 이용할 수 있다.
-관리자는 비디오 재고 파악, 비디오 주문, 매출 파악, 회원 관리를 자동으로 할 수 있다.
-관리자는 회원의 연체료, 비디오 손상에 따른 연체료 등을 관리 할 수 있다.
-오프라인에서의 인건비 감소, 효율적인 비디오, 회원 관리

1.2 개발 내용

사용자	개발 내용
고객	1) 회원가입: 사용자의 기본 정보를 입력하고, 대여자 정보를 저장해서 비디오 가게 이용의 권한을 부여한다.(회원 카드를 발급 받는다.) 2) 로그인: 회원가입 시 저장된 DB에서 고객의 정보를 불러와 일치하면 로그인 성공 3) 비디오 검색: 소비자가 원하는 비디오의 정보를 검색하여 조회한다.(비디오 출시일, 감독 등) 4) 대여하기: 원하는 제품을 구매할 때 사용하는 결제 수단을 선택하고 주문 DB목록에 추가한다. 5) 대여내역, 연체 확인: 소비자가 대여했던 비디오를 조회 할 수 있고 연체상황을 알 수 있다.
직원	1) 주문: 비디오 주문을 할 수 있다. 2) 납품: 비디오 납품을 관리한다. 3) 품질: 비디오 품질과 정보를 관리 한다. 4) 고객관리: 고객 정보 수정, 삭제, 갱신, 목록 출력 등 고객 정보를 관리한다. 5) 대여/반납관리: 고객의 대여/반납 현황, 연체 관리를 한다. 6) 관리자 매출관리: 관리자 직급의 직원은 비디오가게의 매출을 관리한다.
공급업체	1) 공급업체 관리: 서버 관리자로부터 공급업체들을 관리한다. 2) 업체별 공급제품관리: 업체별로 공급하는 비디오가 다르기 때문에 나

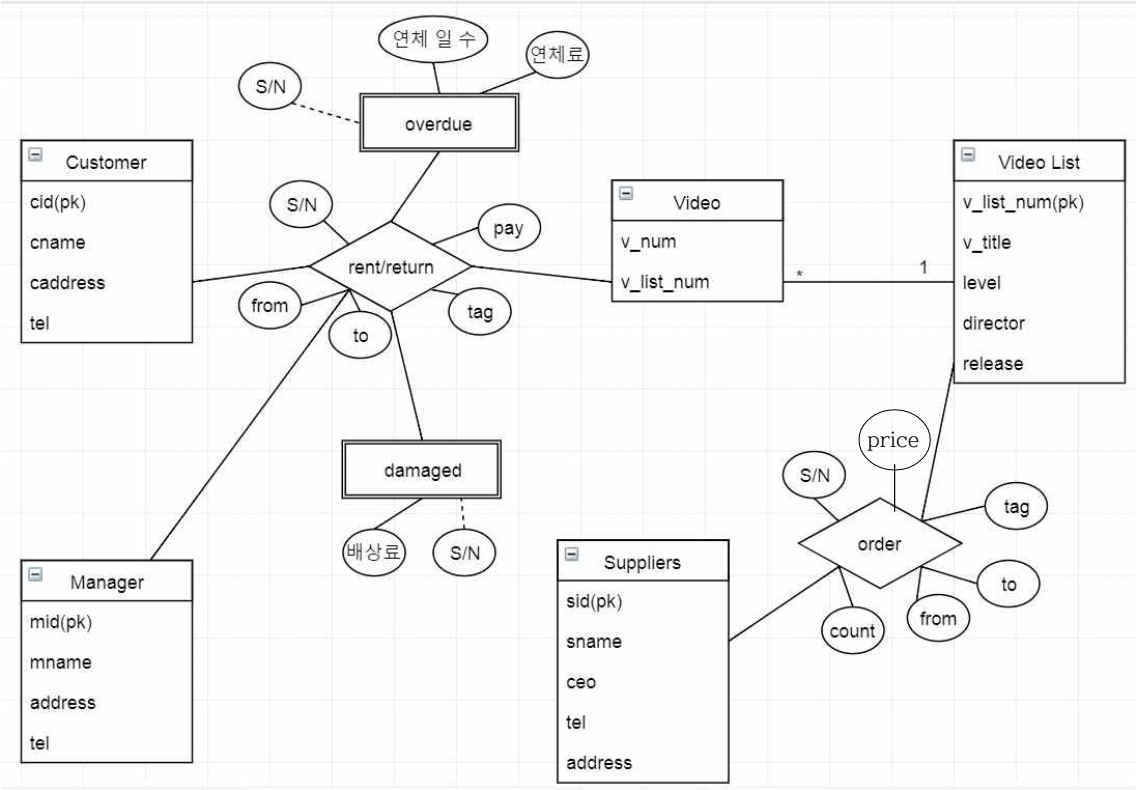
	누어서 관리한다.
비디오	1) 비디오를 등록: 직원은 새로운 비디오 정보를 DB에 등록한다 2) 비디오 검색: 직원, 소비자는 원하는 비디오를 검색 할 수 있다. 3) 비디오 삭제: 존재하지 않는 비디오 정보를 삭제한다.
대여, 반납	1) 대여: 비디오 대여 내역을 관리 할 수 있다. 2) 반납: 비디오 반납 여부를 알 수 있다. 3) 연체: 연체자 정보를 알 수 있다. 4) 파손 관리: 파손정도, 파손된 비디오, 파손 원인 등을 관리한다. 상황에 따라 파손 배상료 지불 여부를 알 수 있다. 5) 연체자 목록 출력: 연체된 회원의 정보를 알 수 있다.

1.3 Use-Case Diagram



2. 데이터 설계

2.1 Class-Diagram



2.2 릴레이션 스키마 테이블

고객(customer)				
고객 번호	cid	INT	50	PK
고객명	cname	CHAR	50	NN
전화번호	ctel	CHAR	50	NN
주소	caddress	CHAR	100	NN
이메일	cemail	CHAR	50	NN

직원(Manager)				
직원번호	mid	ING	10	PK
직원명	mname	CHAR	50	NN
직원주소	maddress	CHAR	100	NN
전화번호	mtel	CHAR	10	NN

비디오목록(VideoList)				
비디오 리스트번호	video_list_num	INT	10	PK
비디오명	vname	CHAR	50	NN
비디오 출시 일	vrelease	CHAR	10	NN
비디오 대여가	vprice	CHAR	50	NN
감독 이름	vdirector	CHAR	50	NN

비디오(Video)				
비디오 번호	vnum	INT	50	PK
비디오 리스트번호	video_list_num	INT	50	NN
상태 tag	vtag	INT	10	NN

연체(Overdue)				
비디오 번호	vnum	INT	10	PK
고객 아이디	cid	CHAR	50	PK
비디오명	vname	CHAR	50	NN
연체일	overdue	CHAR	50	NN
연체료	overcharge	CHAR	50	NN

손상(Damaged)				
손상 번호	did	INT	10	PK
고객 아이디	cid	CHAR	50	NN
비디오 번호	vnum	INT	10	NN
대여 번호	rnum	INT	10	NN
배상료	dfee	CHAR	50	NN

대여/반납 하다(Rent/Return)				
대여번호	rnum	INT	10	PK
고객아이디	cid	INT	10	NN
비디오 번호	vnum	INT	10	NN
대여날짜	rfrom	CHAR	10	NN
반납날짜	rto	CHAR	50	NN
요금	rpay	CHAR	50	NN
대여/반납 tag	rtag	CHAR	10	NN

주문하다(Order)				
주문 번호	onum	CHAR	50	PK
비디오 번호	vlistnum	CHAR	50	PK
업체번호	sid	CHAR	50	NN
업체 명	sname	CHAR	50	NN
수량	count	INT	10	NN
단가	price	INT	30	NN
주문/납품 tag	otag	CHAR	10	NN

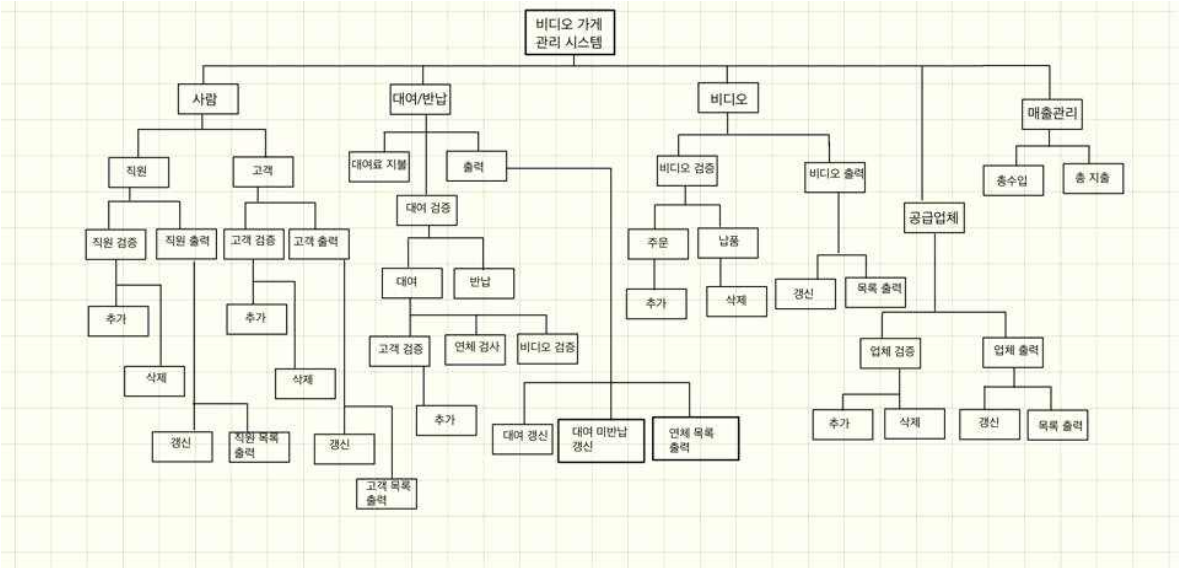
공급업체(Supplier)				
업체번호	sid	INT	10	PK
업체명	sname	CHAR	50	NN
업체ceo	sceo	CHAR	50	NN
업체주소	saddress	CHAR	100	NN
전화번호	stel	CHAR	20	NN

2.3 자료구조 설계

클래스	내용
Costomer	cid : int cname : string caddress : string ctel : string ceamil : string
Manager	mid : int mname : string maddress : string mtel : string memail : string
Supplier	SID :int SNumber :int SCEO : string Saddress : string S_tel : string
Video	vnum : int v_list_num : int
Video List	v_list_num : int vname : string vdirector : string vrelease : string vsupplier : string vprice : string vamount : int

3. 시스템 아키텍처 설계

3.1 아키텍처 다이어그램



3.2 HW spec

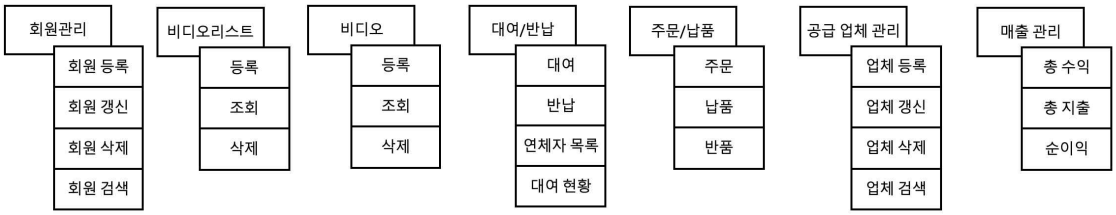
- 데이터베이스를 구축할 서버 컴퓨터
- 인터넷이 가능한 PC
- 모바일 어플리케이션을 구동할 수 있는 스마트폰

3.3 SW spec

- Ms windows 7 이상
- 안드로이드 젤리빈 이상
- Java, mysql 등을 활용한 개발 환경
- html, css, js 등을 활용한 웹사이트 환경

4. 사용자 인터페이스(UI) 설계

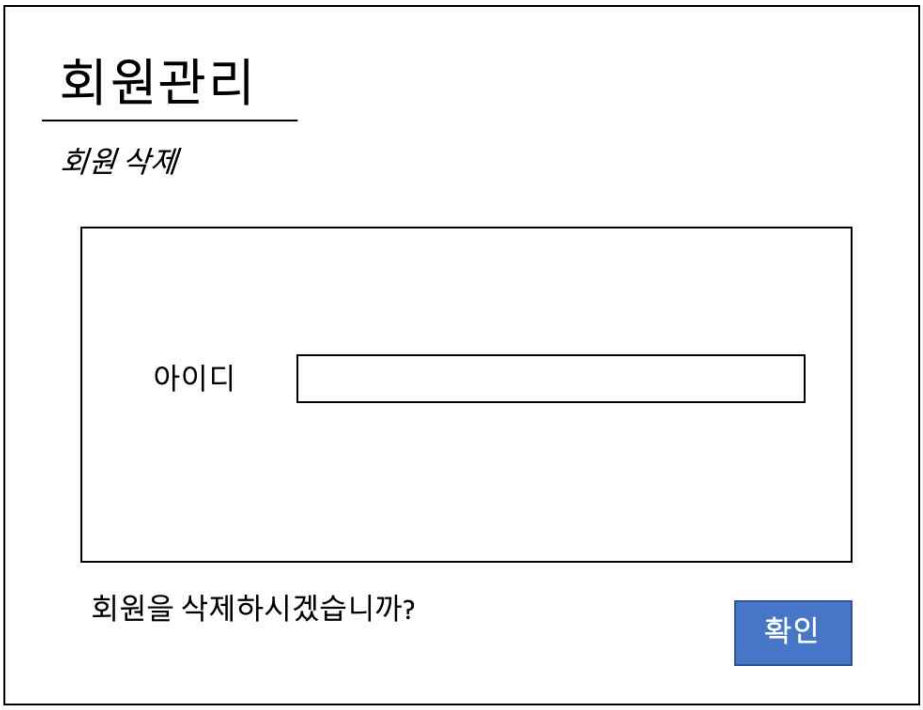
4.1 메인 메뉴



4.2 회원 관리 메뉴


회원관리
회원 갱신
이름
홍길동
주소
서울시 마포구
전화번호
01049384958
수정할 정보를 입력하세요.
확인


회원관리
회원 조회
아이디
조회할 아이디를 입력하세요.
확인


회원관리
회원 삭제
아이디
회원을 삭제하시겠습니까?
확인

4.3 비디오 리스트 메뉴


비디오 리스트
등록
번호
제목
출시 일
대여 가격
감독
등록할 정보를 입력하세요.
확인


비디오 리스트
삭제
번호
삭제할 번호를 입력하세요.
확인

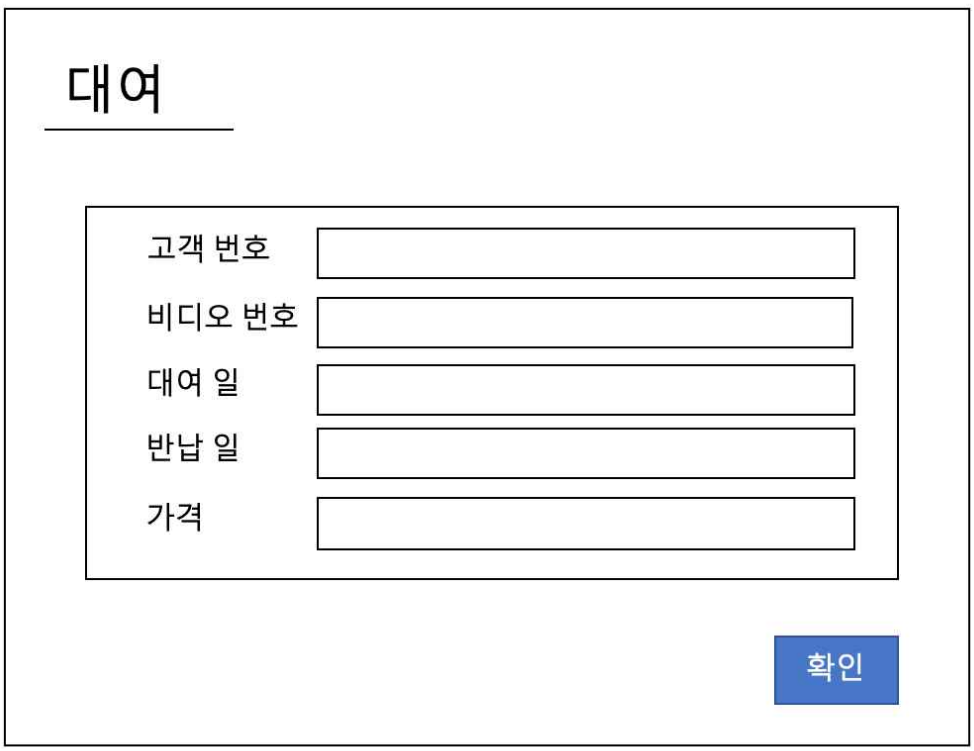

비디오 리스트
조회
번호
제목
출시 일
대여 가격
감독
조회할 번호를 입력하세요.
확인

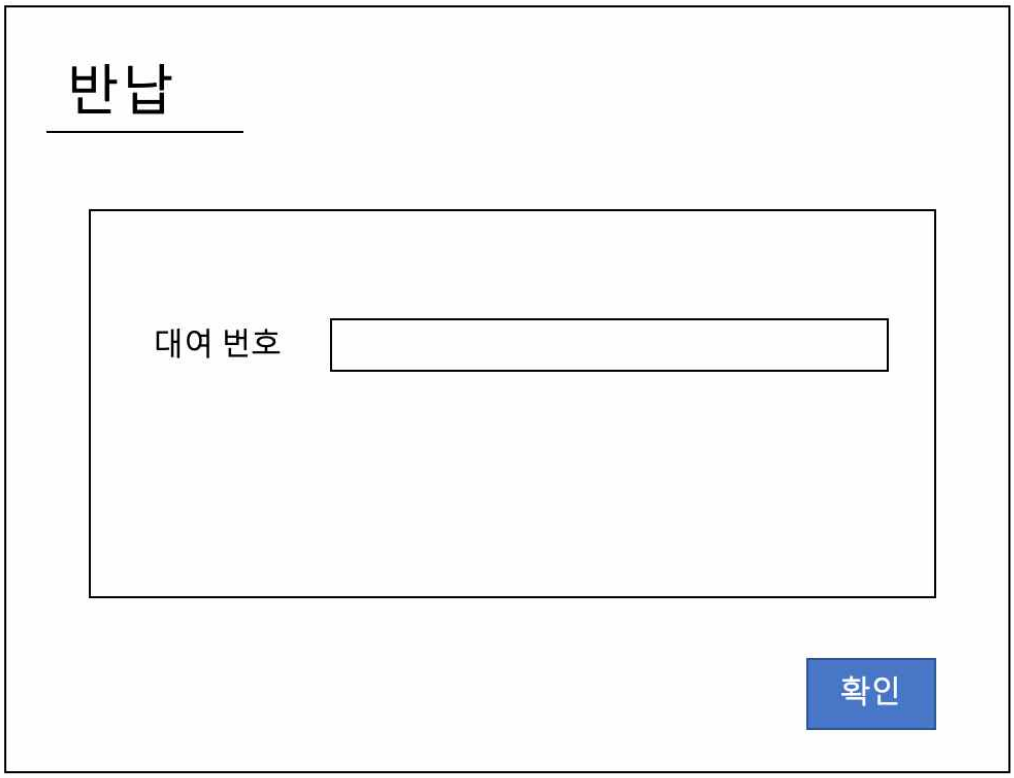
4.4 비디오 메뉴

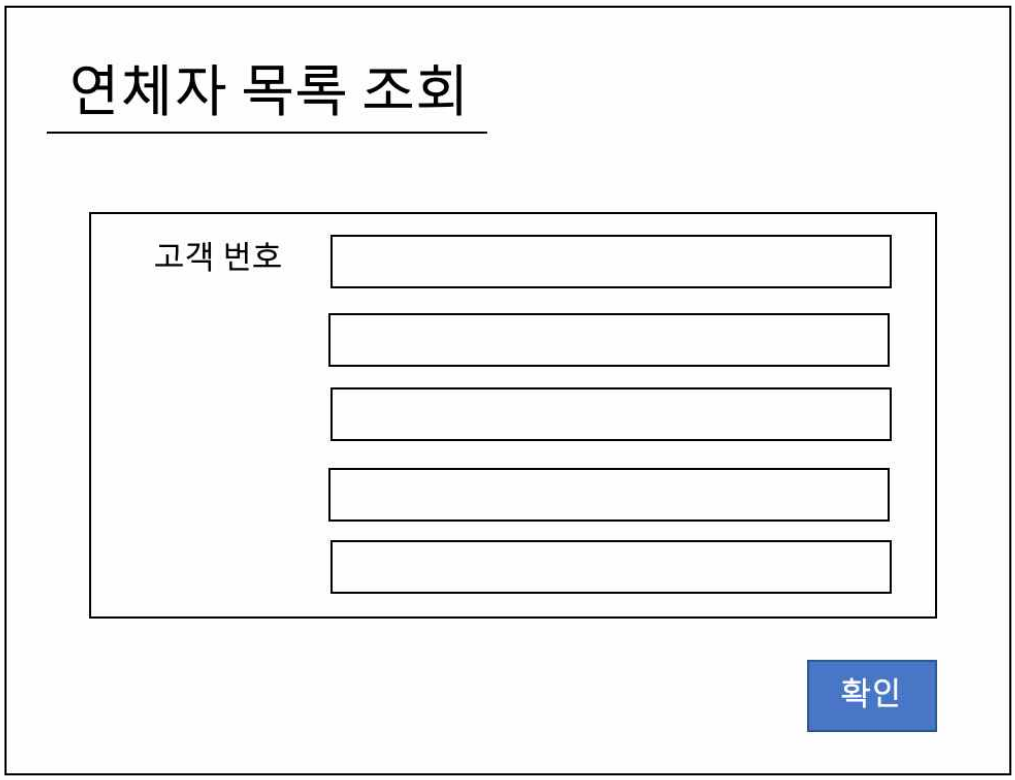

비디오
등록
비디오 번호
비디오 리스트 번호
등록할 정보를 입력하세요.
확인


비디오
삭제
비디오 번호
삭제할 번호를 입력하세요.
확인

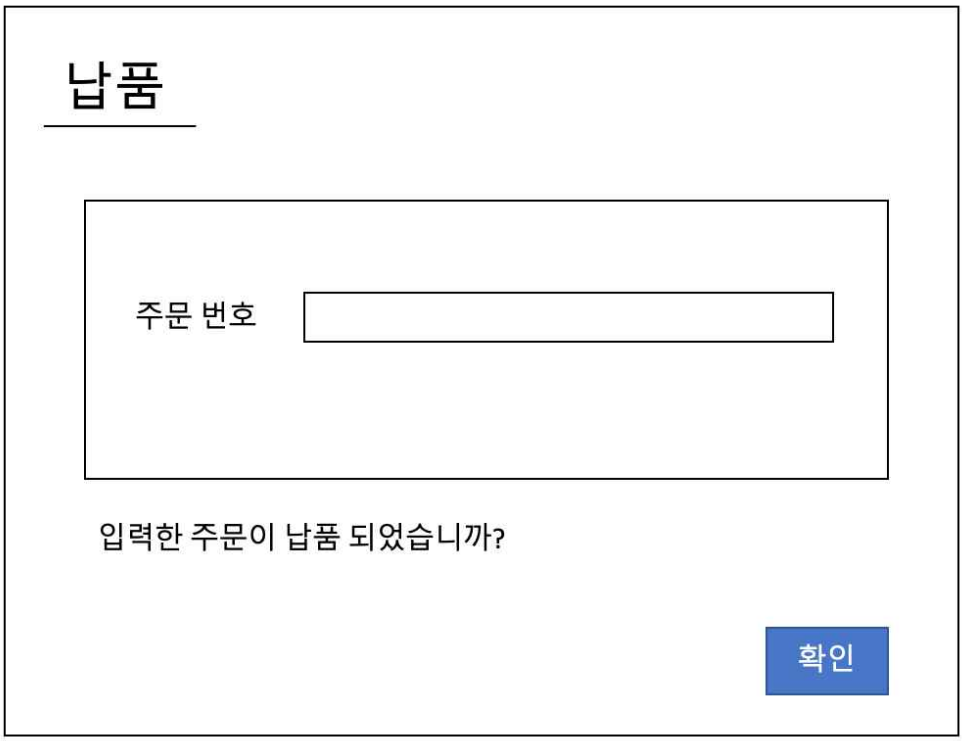
4.5 대여/반납 메뉴




반납
대여 번호
확인


연체자 목록 조회
고객 번호
확인

4.6 주문/반품 메뉴

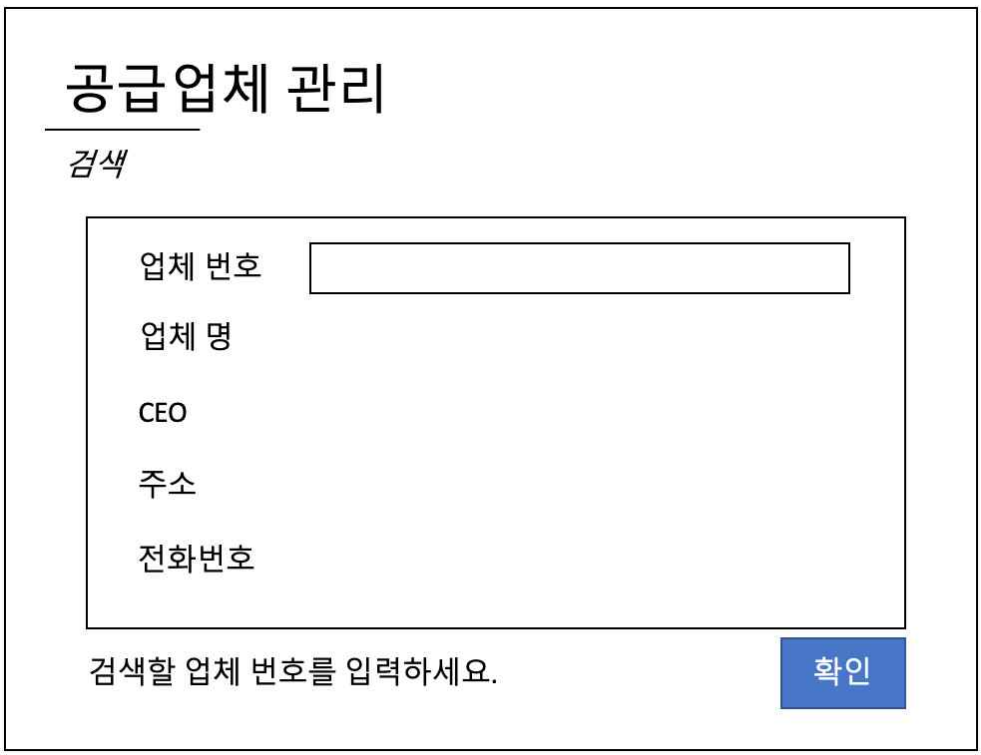


4.7 공급 업체 관리 메뉴

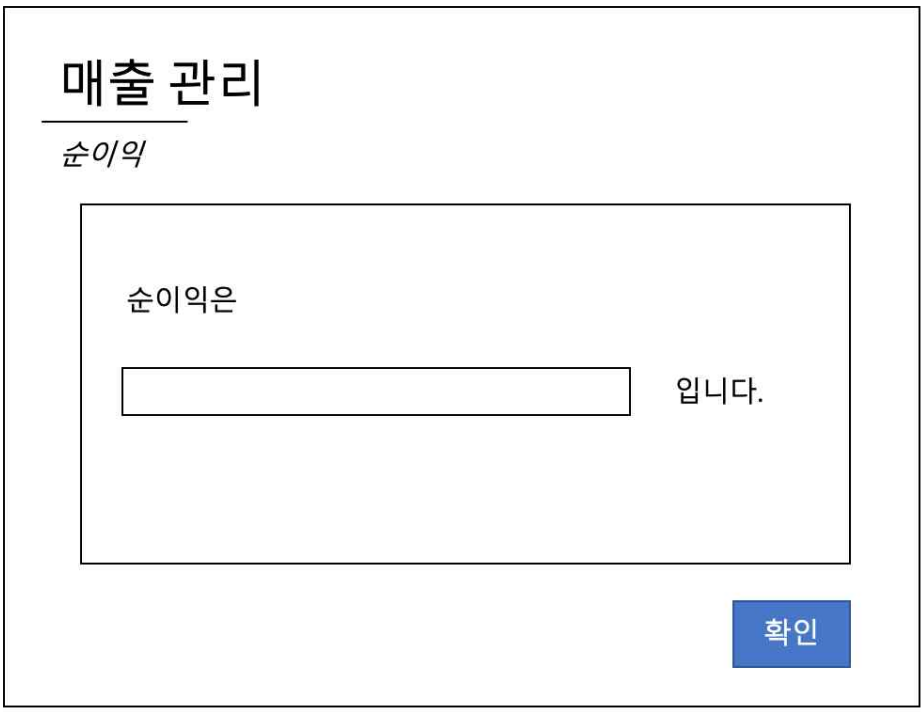

공급업체 관리
등록
업체 번호
업체 명
CEO
주소
전화번호
등록할 업체 정보를 입력하세요.
확인


공급업체 관리
갱신
업체 번호
업체 명
CEO
주소
전화번호
수정할 업체 정보를 입력하세요.
확인


공급업체 관리
삭제
업체 번호
삭제할 업체 번호를 입력하세요.
확인

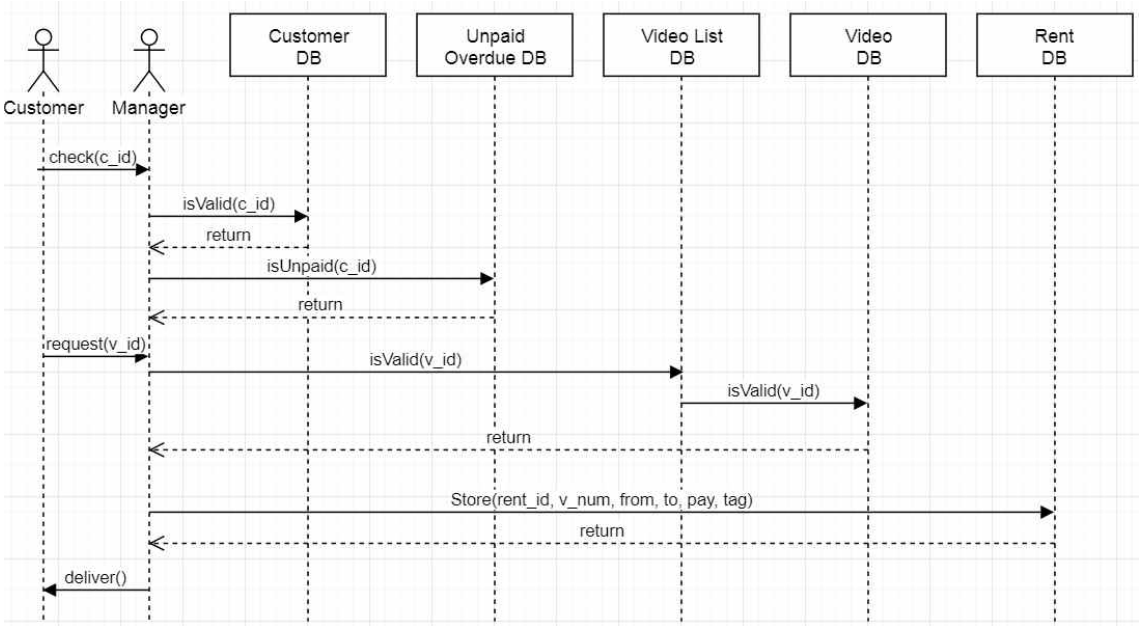

공급업체 관리
검색
업체 번호
업체 명
CEO
주소
전화번호
검색할 업체 번호를 입력하세요.
확인

4.8 매출 관리

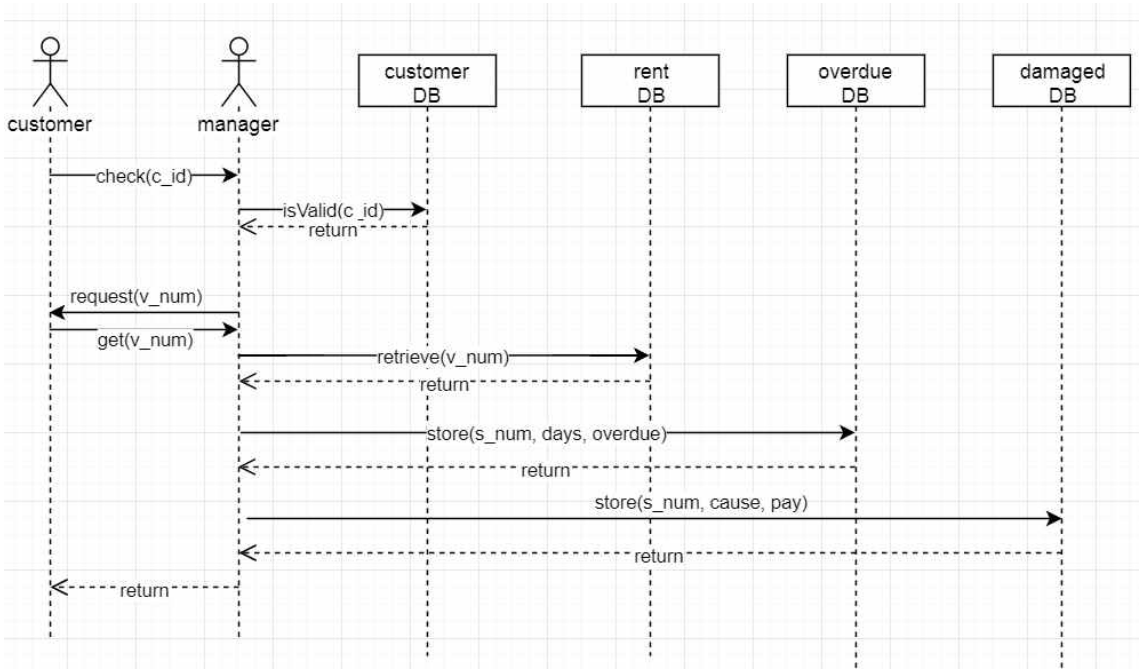

매출 관리
순이익
순이익은
입니다.
확인

5. 시퀀스다이어그램의 메소드 알고리즘 설계

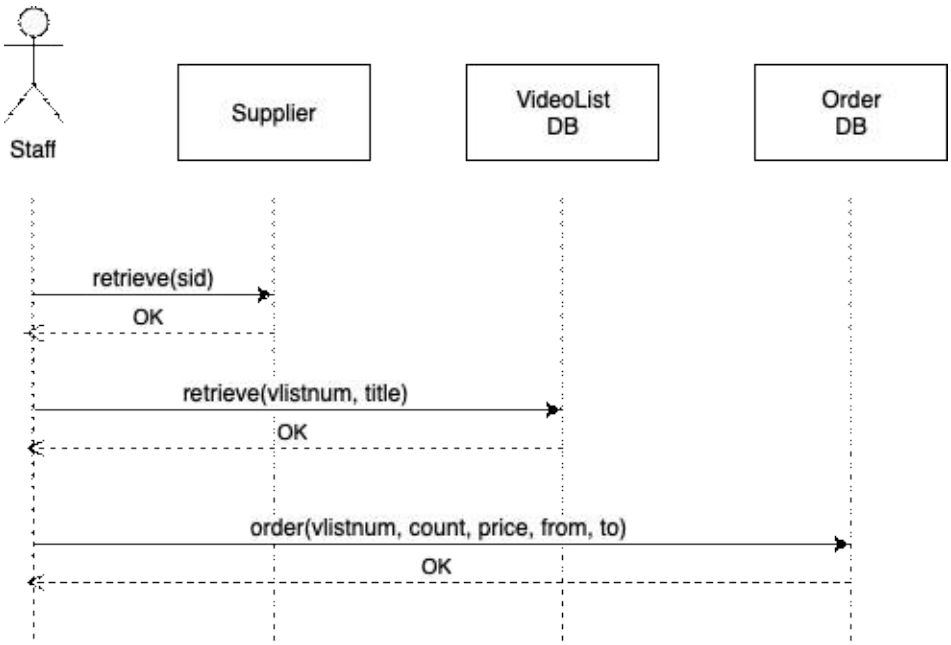
5.1 Use-Case '대여하다'



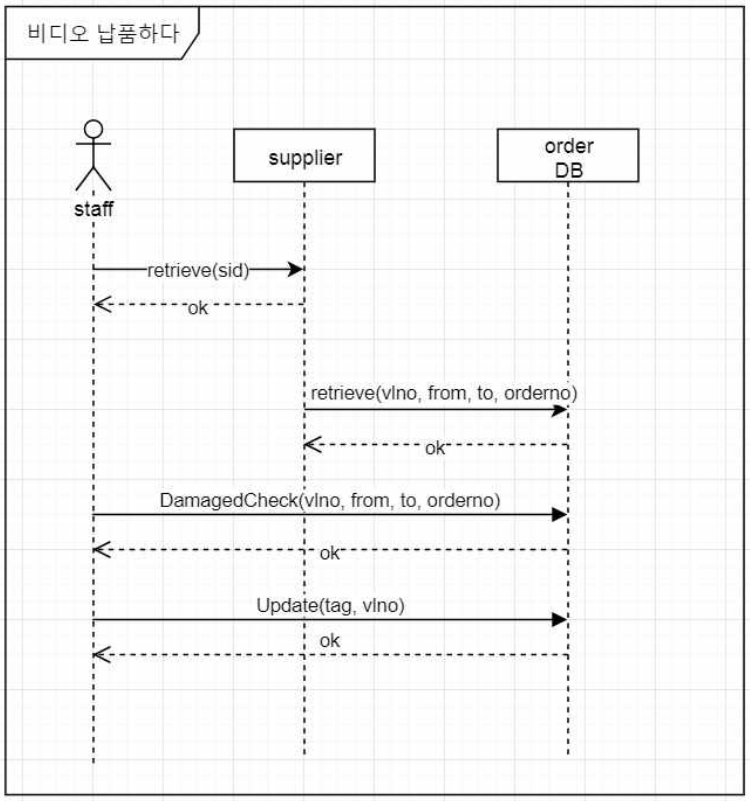
5.2 Use-Case '반납하다'



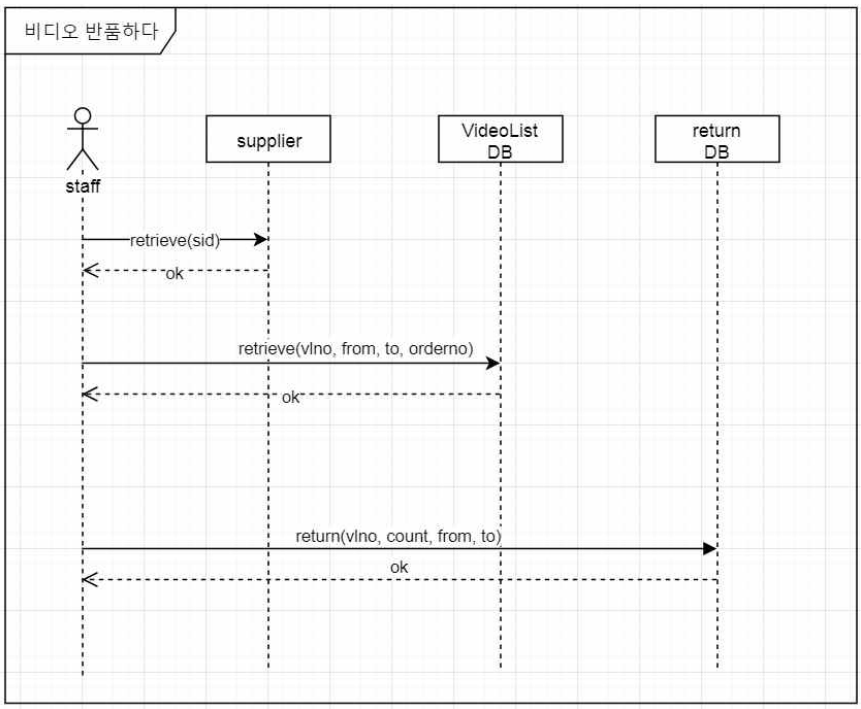
5.3 Use-Case '주문하다'



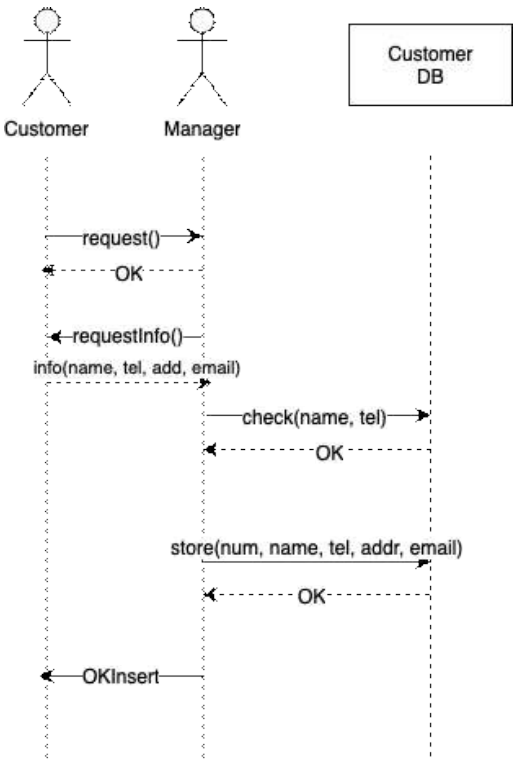
5.4 Use-Case '납품하다'



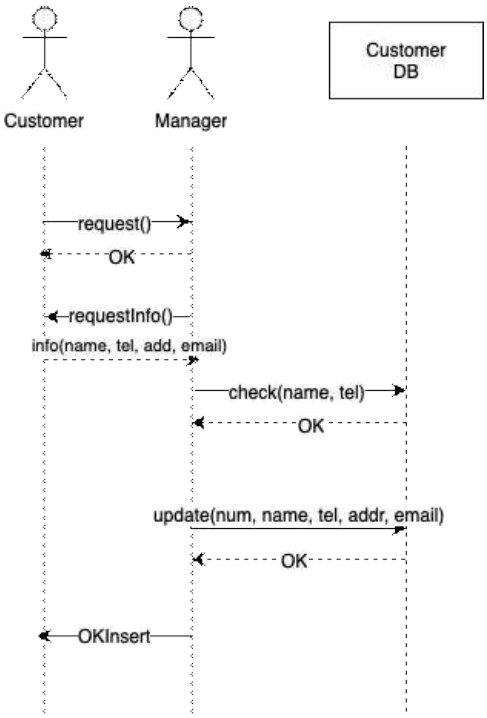
5.5 Use-Case '반품하다'



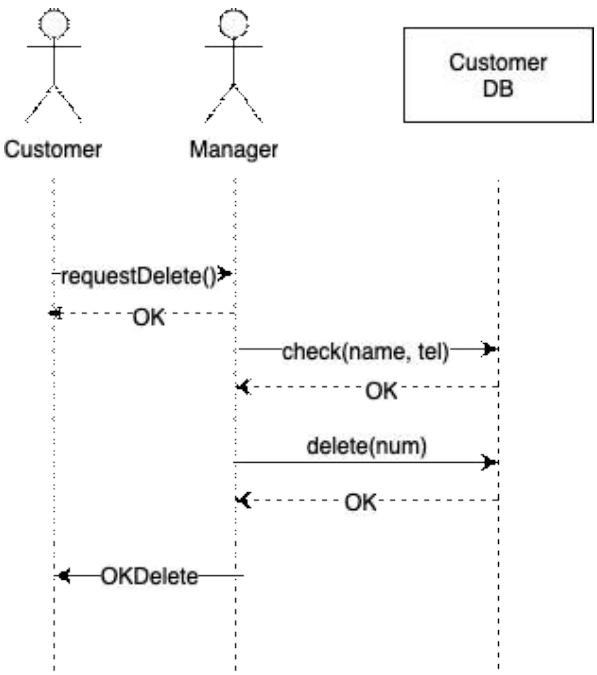
5.4 Use-Case '회원 등록하다'



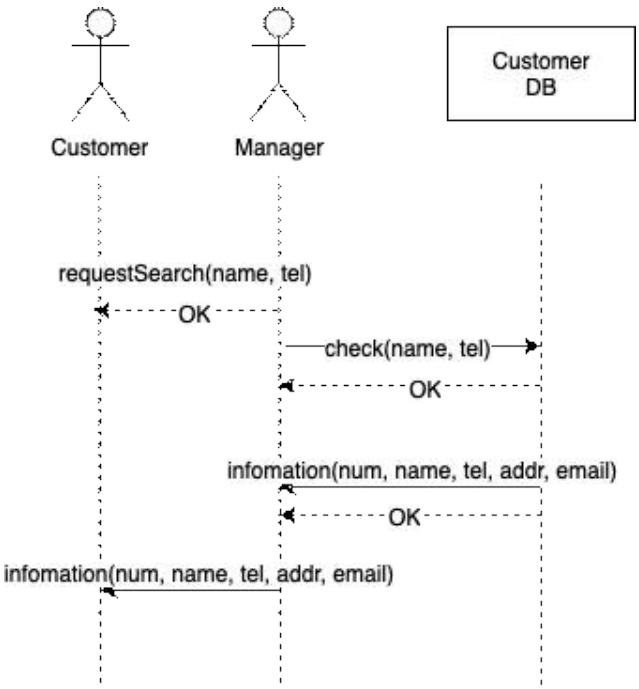
5.5 Use-Case '회원 갱신하다'



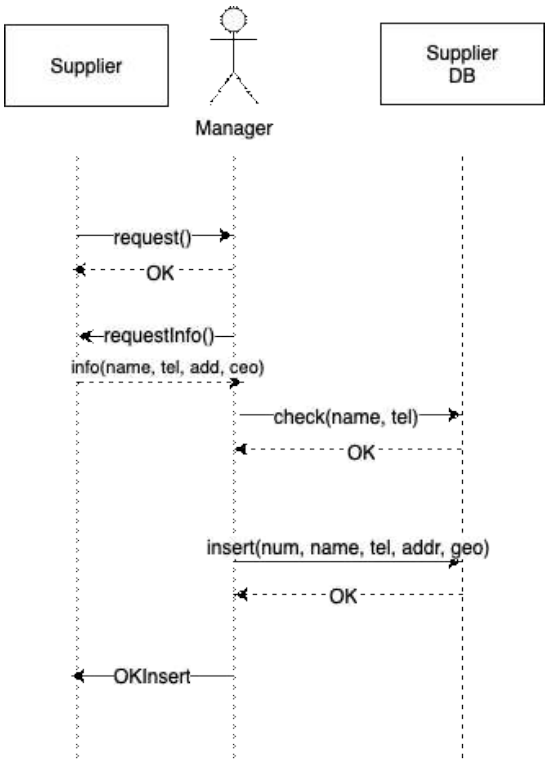
5.6 Use-Case '회원 삭제하다'



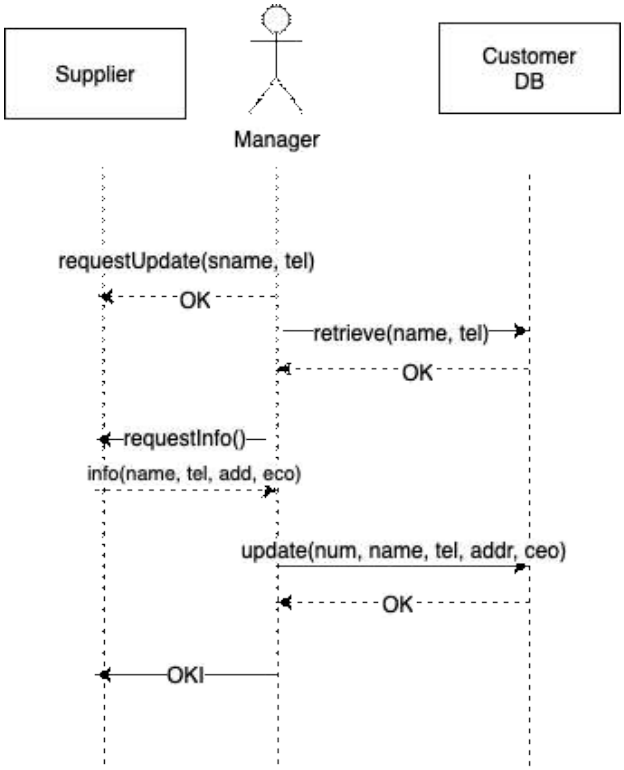
5.7 Use_Case '회원 조회하다'



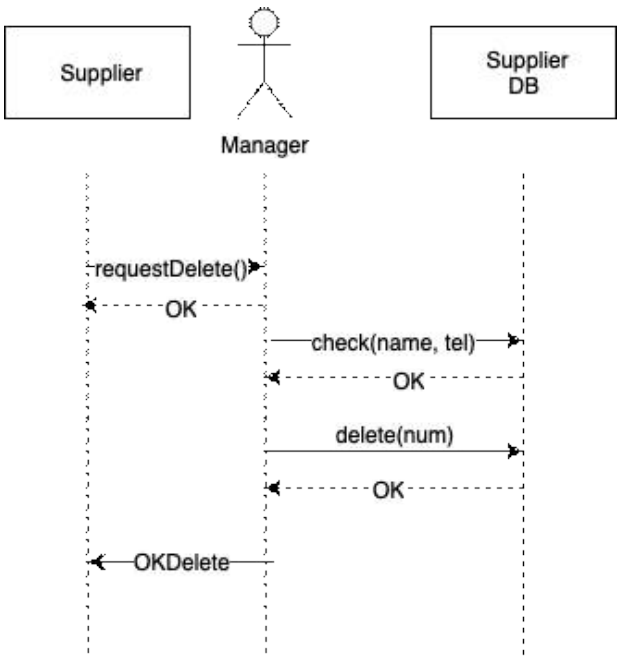
5.8 Use-Case '공급업체 등록하다'



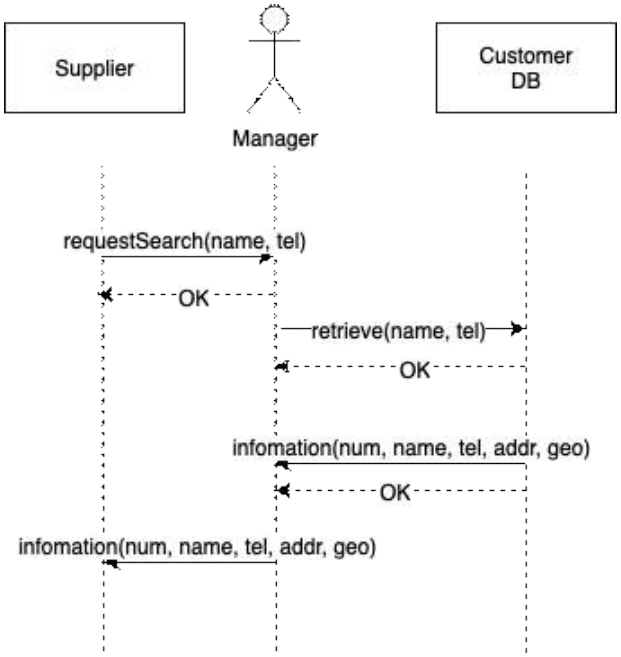
5.9 Use-Case '공급업체 갱신하다'



5.10 Use-Case '공급업체 삭제하다'



5.11 Use-Case '공급업체 조회하다'



6. Method 알고리즘 설계

6.1 class Customer

6.1.1 insert()

인터페이스	public void customerInsert(cid, cname, caddress, ctel)
기능	고객 데이터베이스에 고객 정보를 저장한다.

알고리즘	public void customerInsert(cid, cname, caddress, ctel)
	{
	//고객 정보에 저장할 변수를 생성하고 새로 입력되는 값들을 대입한다.
	private String cid=cid; //고객 아이디
	private String cname=cname; //고객 이름
	private String caddress=caddress; //고객 주소
	private String ctel=ctel; //고객 전화번호
	
	//고객 정보를 데이터베이스에 연결하여 저장하기 위한 url 변수를 생성
	String sql;
	
	//데이터베이스에 고객 정보를 추가하기 위한 쿼리문
	sql = “insert into customer(cid, cname, caddress, ctel) values(?,?,?,?)”;
	
	//데이터베이스와 연결 한 후 추가된 고객정보와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, cid); //고객 아이디를 db에 저장
	pstmt.setString(2, cname); //고객 이름을 db에 저장
	pstmt.setString(3, caddress); //고객 주소를 db에 저장
	pstmt.setString(4, ctel); //고객 전화번호를 db에 저장
	
	}

6.1.2 update()

인터페이스	public void customerUpdate(cid, cname, caddress, ctel)
기능	고객 데이터베이스에 저장된 고객 정보를 수정한다.

알고리즘	public void customerUpdate(cid, cname, caddress, ctel)
	{
	//수정할 고객정보의 아이디, 이름, 주소, 전화번호에 해당하는 변수를 생성한다.
	private String cid=cid; //고객 아이디
	private String cname=cname; //고객 이름
	private String caddress=caddress; //고객 주소

	private String ctel=ctel; //고객 전화번호
	
	//데이터베이스에 연결할 쿼리문을 작성하기 위한 변수를 생성한다.
	String sql;
	
	//고객 정보 업데이트를 하기 위한 쿼리문을 작성한다.
	sql= “update customer set cid=?, cname=?, caddress=?, ctel=?”;
	
	//데이터베이스와 연결 한 후 수정된 고객정보와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, cid); //수정할 고객 아이디를 쿼리문에 전송
	pstmt.setString(2, cname); //수정할 고객 이름을 쿼리문에 전송
	pstmt.setString(3, caddress); //수정할 고객 주소를 쿼리문에 전송
	pstmt.setString(4, ctel); //수정할 고객 전화번호를 쿼리문에 전송
	}

6.1.3 delete()

인터페이스	public void customerDelete(cid)
기능	고객 데이터베이스에 저장된 고객 정보를 삭제한다.

알고리즘	public void customerDelete(cid)
	{
	//삭제할 고객의 아이디를 저장할 변수를 생성한다.
	String cid=cid;
	//데이터베이스에 쿼리명령을 보내기 위한 sql 변수를 생성한다.
	Stirng sql;
	
	//해당 고객아이디를 데이터베이스에서 삭제하기 위한 쿼리문을 작성한다.
	sql = “delete from customer where cid=?”;
	
	//데이터베이스와 연결 한 후 삭제할된 고객아이디와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, cid);
	}

6.1.4 retrieve()

인터페이스	public void customerRetrieve(cid)
기능	고객 데이터베이스에 저장된 고객 정보를 검색한다.
알고리즘	public void customerRetrieve(cid) {

	//검색할 고객의 아이디를 저장할 변수를 생성한 후 대입한다.
	String cid=cid;
	//데이터베이스에 보낼 쿼리문을 담을 변수를 생성한다.
	String sql;
	
	//고객 정보를 검색하기 위한 쿼리문을 작성한다.
	sql= “select cid, cname, caddress, ctel from customer where cid=?”;
	
	//데이터베이스에 연결 한 후 검색할 고객의 아이디 값을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, cid);
	
	}

6.2 class Employee

6.2.1 insert()

인터페이스	public void empInsert(mid, mname, maddress, mtel)
기능	직원 데이터베이스에 직원 정보를 저장한다.

알고리즘	public void empInsert(mid, mname, maddress, mtel)
	{
	//직원 정보에 저장할 변수를 생성하고 새로 입력되는 값들을 대입한다.
	private String mid=mid; //직원 아이디
	private String mname=mname; //직원 이름
	private String maddress=maddress; //직원 주소
	private String mtel=mtel; //직원 전화번호
	
	//직원 정보를 데이터베이스에 연결하여 저장하기 위한 url 변수를 생성
	String sql;
	
	//데이터베이스에 직원 정보를 추가하기 위한 쿼리문
	sql = “insert into employee(mid, mname, maddress, mtel) values(?,?,?,?)”;
	
	//데이터베이스와 연결 한 후 추가된 직원 정보와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, mid); //직원 아이디를 db에 저장
	pstmt.setString(2, mname); //직원 이름을 db에 저장
	pstmt.setString(3, maddress); //직원 주소를 db에 저장
	pstmt.setString(4, mtel); //직원 전화번호를 db에 저장
	
	}

6.2.2 update()

인터페이스	public void empUpdate(mid, mname, maddress, mtel)
기능	직원 데이터베이스에 저장된 직원 정보를 수정한다.

알고리즘	public void empUpdate(mid, mname, maddress, mtel)
	{
	//수정할 직원 정보의 아이디, 이름, 주소, 전화번호에 해당하는 변수를 생성한다.
	private String mid=mid; //직원 아이디
	private String mname=mname; //직원 이름
	private String maddress=maddress; //직원 주소
	private String mtel=mtel; //직원 전화 번호

	//데이터베이스에 연결할 쿼리문을 작성하기 위한 변수를 생성한다.
	String sql;
	
	//직원 정보 업데이트를 하기 위한 쿼리문을 작성한다.
	sql= “update employee set mid=?, mname=?, maddress=?, mtel=?”;
	
	//데이터베이스와 연결 한 후 수정된 직원 정보와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, mid); //수정할 직원 아이디 쿼리문에 전송
	pstmt.setString(2, mname); //수정할 직원 이름 쿼리문에 전송
	pstmt.setString(3, maddress); //수정할 직원 주소 쿼리문에 전송
	pstmt.setString(4, mtel); //수정할 직원 전화 번호 쿼리문에 전송
	}

6.2.3 delete()

인터페이스	public void empDelete(mid)
기능	직원 데이터베이스에 저장된 직원 정보를 삭제한다.

알고리즘	public void empDelete(mid)
	{
	//삭제할 직원의 아이디를 저장할 변수를 생성한다.
	String mid=mid;
	//데이터베이스에 쿼리명령을 보내기 위한 sql 변수를 생성한다.
	Stirng sql;
	
	//해당 직원 아이디를 데이터베이스에서 삭제하기 위한 쿼리문을 작성한다.
	sql = “delete from employee where mid=?”;
	
	//데이터베이스와 연결 한 후 삭제할 직원아이디와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, mid);
	}

6.2.4 retrieve()

인터페이스	public void empRetrieve(mid)
기능	직원 데이터베이스에 저장된 직원 정보를 검색한다.

알고리즘	public void empRetrieve(mid)
	{
	//검색할 고객의 아이디를 저장할 변수를 생성한 후 대입한다.
	String mid=mid;

	//데이터베이스에 보낼 쿼리문을 담을 변수를 생성한다.
	String sql;
	
	//고객 정보를 검색하기 위한 쿼리문을 작성한다.
	sql= “select mid, mname, maddress, mtel from employee where mid=?”;
	
	//데이터베이스에 연결 한 후 검색할 직원 아이디 값을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, mid);
	
	}

6.3 class Video

6.3.1 insert()

인터페이스	public void videoInsert(vnum, vlistnum, tag)
기능	비디오 데이터베이스에 비디오 정보를 저장한다.

알고리즘	public void videoInsert(vnum, vlistnum, tag)
	{
	//비디오 정보에 저장할 변수를 생성하고 새로 입력되는 값들을 대입한다.
	private String vnum = vnum; //비디오 번호
	private String vlistnum = vlistnum; //비디오 리스트 번호
	private String tag = tag; //비디오 상태 태그
	
	//비디오 정보를 데이터베이스에 연결하여 저장하기 위한 url 변수를 생성
	String sql;
	
	//데이터베이스에 비디오 정보를 추가하기 위한 쿼리문
	sql = “insert into video(vnum, vlistnum, tag) values(?,?,?)”;
	
	//데이터베이스와 연결 한 후 추가된 비디오 정보와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, vnum); //비디오 번호를 db에 저장
	pstmt.setString(2, vlistnum); //비디오 리스트 번호를 db에 저장
	pstmt.setString(3, tag); //비디오 상태 태그를 db에 저장
	}

6.3.2 update()

인터페이스	public void videoUpdate(vnum, vlistnum, tag)
기능	비디오 데이터베이스에 저장된 비디오를 수정한다.

알고리즘	public void videoUpdate(vnum, vlistnum, tag)
	{
	//수정할 비디오 정보의 비디오번호, 비디오리스트번호에 해당하는 변수를 생성한다.
	private String vnum = vnum; //비디오 번호
	private String vlistnum = vlistnum; //비디오 리스트 번호
	private String tag = tag; //비디오 상태 태그
	
	//데이터베이스에 연결할 쿼리문을 작성하기 위한 변수를 생성한다.
	String sql;
	
	//비디오 정보 업데이트를 하기 위한 쿼리문을 작성한다.

	sql= “update video set vnum=?, vlistnum=?, tag=?”;
	
	//데이터베이스와 연결 한 후 수정된 비디오 정보와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, vnum); //수정할 비디오 번호 쿼리문에 전송
	pstmt.setString(2, vlistnum); //수정할 비디오 리스트 번호 쿼리문에 전송
	pstmt.setString(3, tag); //수정할 비디오 리스트 상태 태그 쿼리문에 전송
	}

6.3.3 delete()

인터페이스	public void videoDelete(vnum, vlistnum, tag)
기능	비디오 데이터베이스에 저장된 비디오를 삭제한다.

알고리즘	public void videoDelete(vnum, vlistnum, tag)
	{
	//삭제할 비디오번호를 저장할 변수를 생성한다.
	String vnum = vnum;
	//데이터베이스에 쿼리명령을 보내기 위한 sql 변수를 생성한다.
	Stirng sql;
	
	//해당 직원 아이디를 데이터베이스에서 삭제하기 위한 쿼리문을 작성한다.
	sql = “delete from video where vnum=?”;
	
	//데이터베이스와 연결 한 후 삭제할 비디오번호와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, vnum);
	}

6.3.4 retrieve()

인터페이스	public void videoRetrieve(vnum)
기능	비디오 데이터베이스에 저장된 비디오 정보를 검색한다.

알고리즘	public void videoRetrieve(vnum)
	{
	//검색할 비디오번호를 저장할 변수를 생성한 후 대입한다.
	String vnum=vnum;
	//데이터베이스에 보낼 쿼리문을 담을 변수를 생성한다.
	String sql;
	
	//비디오 정보를 검색하기 위한 쿼리문을 작성한다.

	sql= “select vnum. vlistnum from video where vnum=?”;
	
	//데이터베이스에 연결 한 후 검색할 비디오번호 값을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, vnum);
	}

6.4 class VideoList

6.4.1 insert()

인터페이스	public void videoListInsert(vlistnum, vname, vrelease, vprice, vamount, vdirector)
기능	비디오 리스트 데이터베이스에 비디오 리스트 정보를 저장한다.

알고리즘	public void videoListInsert(vlistnum, vname, vrelease, vprice, vamount, vdirector)
	{
	//비디오 리스트 정보에 저장할 변수를 생성하고 새로 입력되는 값들을 대입한다.
	private String vlistnum = vlistnum; //비디오 목록 번호
	private String vname = vname; //비디오 이름
	private String vrelease = vrelease; //비디오 출판일
	private String vprice = vprice; //비디오 가격
	private String vamount = vamount; //비디오 수량
	private String vdirector = vdirector; //비디오 감독
	
	//비디오 리스트 정보를 데이터베이스에 연결하여 저장하기 위한 url 변수를 생성
	String sql;
	
	//데이터베이스에 비디오 리스트 정보를 추가하기 위한 쿼리문
	sql = “insert into videolist(vlistnum, vname, vrelease, vprice, vamount, vdirector) values(?,?,?,?,?,?)”;
	
	//데이터베이스와 연결 한 후 추가된 비디오 정보와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, vlistnum); //비디오 목록 번호를 db에 저장
	pstmt.setString(2, vname); //비디오 이름을 db에 저장
	pstmt.setString(3, vrelease); //비디오 출판일 db에 저장
	pstmt.setString(4, vprice); //비디오 가격 db에 저장
	pstmt.setString(5, vamount); //비디오 수량 db에 저장
	pstmt.setString(6, vdirector); //비디오 감독 db에 저장
	}

6.4.2 update()

인터페이스	public void videoListUpdate(vlistnum, vname, vrelease, vprice, vamount, vdirector)
기능	비디오 리스트 데이터베이스에 비디오 리스트 정보를 수정한다.
알고리즘	public void videoListUpdate(vlistnum, vname, vrelease, vprice,

	vamount, vdirector)
	{
	//수정할 비디오 리스트 정보의 비디오리스트번호, 비디오이름, 개봉일, 가격, 수량, 감독에 해당하는 변수를 생성한다.
	private String vlistnum = vlistnum; //비디오 목록 번호
	private String vname = vname; //비디오 이름
	private String vrelease = vrelease; //비디오 출판일
	private String vprice = vprice; //비디오 가격
	private String vamount = vamount; //비디오수량
	private String vdirector = vdirector; //비디오 감독
	
	//데이터베이스에 연결할 쿼리문을 작성하기 위한 변수를 생성한다.
	String sql;
	
	//비디오 리스트 정보 업데이트를 하기 위한 쿼리문을 작성한다.
	sql= “update videolist set vlistnum=?, vname=?, vrelease=?, vprice=?, vamount=?, vdirector=?”;
	
	//데이터베이스와 연결 한 후 수정된 비디오리스트정보와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, vlistnum); //수정할 비디오 목록 번호 쿼리문에 전송
	pstmt.setString(2, vname); //수정할 비디오 이름 쿼리문에 전송
	pstmt.setString(3, vrelease); //수정할 비디오 출판일 쿼리문에 전송
	pstmt.setString(4, vprice); //수정할 비디오 가격 쿼리문에 전송
	pstmt.setString(5, vamount); //수정할 비디오 수량 쿼리문에 전송
	pstmt.setString(6, vdirector); //수정할 비디오 감독 쿼리문에 전송
	}

6.4.3 delete()

인터페이스	public void videoListDelete(vlistnum)
기능	비디오 리스트 데이터베이스에 비디오 리스트 정보를 삭제한다.

알고리즘	public void videoListDelete(vlistnum)
	{
	//삭제할 비디오리스트번호를 저장할 변수를 생성한다.
	String vlistnum = vlistnum;
	//데이터베이스에 쿼리명령을 보내기 위한 sql 변수를 생성한다.
	Stirng sql;
	
	//해당 비디오 리스트 아이디를 데이터베이스에서 삭제하기 위한 쿼리문을

	작성한다.
	sql = “delete from videolist where vlistnum=?”;
	
	//데이터베이스와 연결 한 후 삭제할 비디오리스트번호와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, vlistnum); //삭제할 비디오리스트번호를 쿼리문에전송
	}

6.4.4 retrieve()

인터페이스	public void videoListRetrieve(vlistnum)
기능	비디오 리스트 데이터베이스에 비디오 리스트 정보를 검색한다.

알고리즘	public void videoListRetrieve(vlistnum)
	{
	//검색할 비디오리스트번호를 저장할 변수를 생성한 후 대입한다.
	String vlistnum=vlistnum;
	
	//데이터베이스에 보낼 쿼리문을 담을 변수를 생성한다.
	String sql;
	
	//비디오리스트 정보를 검색하기 위한 쿼리문을 작성한다.
	sql= “select vlistnum, vname, vrelease, vprice, vamount, vdirector from videolist where vlistnum=?”;
	
	//데이터베이스에 연결 한 후 검색할 비디오리스트 번호를 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, vlistnum); //조회할 비디오 리스트 번호를 쿼리문에전송
	}

6.5 class Supplier

6.5.1 insert()

인터페이스	public void supplierInsert(sid, sname, sceo, saddr, stel)
기능	공급업체 데이터베이스에 공급업체 정보를 저장한다.

알고리즘	public void supplierInsert(sid, sname, sceo, saddr, stel)
	{
	//공급업체 정보에 저장할 변수를 생성하고 새로 입력되는 값들을 대입한다.
	private String sid = sid; //공급업체 업체코드

	private String sname = sname; //공급업체 이름
	private String sceo = sceo; //공급업체 ceo이름
	private String saddr = saddr; //공급업체 주소
	private String stel = stel; //공급업체 전화번호
	
	//공급업체 정보를 데이터베이스에 연결하여 저장하기 위한 url 변수를 생성
	String sql;
	
	//데이터베이스에 공급업체 정보를 추가하기 위한 쿼리문
	sql = “insert into supplier(sid, sname, sceo, saddr, stel) values(?,?,?,?,?)”;
	
	//데이터베이스와 연결 한 후 추가된 공급업체 정보와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, sid); //공급업체 사원 코드 번호를 db에 추가한다.
	pstmt.setString(2, sname); //공급업체 명 db에 추가한다.
	pstmt.setString(3, sceo); //공급업체 ceo 이름을 db에 추가한다.
	pstmt.setString(4, saddr); //공급업체 주소를 db에 추가한다.
	pstmt.setString(5, stel); //공급업체 전화번호를 db에 추가한다.
	}

6.5.2 update()

인터페이스	public void supplierUpdate(sid, sname, sceo, saddr, stel)
기능	공급업체 데이터베이스에 저장된 공급업체 정보를 수정한다.

알고리즘	public void supplierUpdate(sid, sname, sceo, saddr, stel)
	{
	//수정할 공급업체 정보의 아이디, 이름, ceo, 주소, 전화번호에 해당하는 변수를 생성한다.
	private String sid = sid; //공급업체 사원 번호
	private String sname = sname; //공급업체 이름
	private String sceo = sceo; //공급업체 CEO이름
	private String saddr = saddr; //공급업체 주소
	private String stel = stel; //공급업체 전화번호
	
	//데이터베이스에 연결할 쿼리문을 작성하기 위한 변수를 생성한다.
	String sql;
	
	//공급업체 정보 업데이트를 하기 위한 쿼리문을 작성한다.
	sql= “update supplier set sid=?, sname=?, sceo=?, saddr=?, stel=?”;

	//데이터베이스와 연결 한 후 수정된 공급업체 정보와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, sid); //db에 저장된 공급업체 코드 번호를 수정한다.
	pstmt.setString(2, sname); //db에 저장된 공급업체 명을 수정한다.
	pstmt.setString(3, sceo); //db에 저장된 공급업체 CEO명을 수정한다.
	pstmt.setString(4, saddr); //db에 저장된 공급업체 주소를 수정한다.
	pstmt.setString(5, stel); //db에 저장된 공급업체 전화번호를 수정한다.
	}

6.5.3 delete()

인터페이스	public void supplierDelete(sid)
기능	공급업체 데이터베이스에 저장된 공급업체 정보를 삭제한다.

알고리즘	public void supplierDelete(sid)
	{
	//삭제할 공급업체 아이디를 저장할 변수를 생성한다.
	String sid = sid;
	//데이터베이스에 쿼리 명령을 보내기 위한 sql 변수를 생성한다.
	Stirng sql;
	
	//해당 공급업체 아이디를 데이터베이스에서 삭제하기 위한 쿼리문을 작성한다.
	sql = “delete from supplier where sid=?”;
	
	//데이터베이스와 연결 한 후 삭제할 공급업체아이디와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, sid);
	}

6.5.4 retrieve()

인터페이스	public void supplierRetrieve(sid)
기능	공급업체 데이터베이스에 저장된 공급업체 정보를 검색한다.

알고리즘	public void supplierRetrieve(sid)
	{
	//검색할 공급업체 아이디를 저장할 변수를 생성한 후 대입한다.
	String sid = sid;
	
	//데이터베이스에 보낼 쿼리문을 담을 변수를 생성한다.
	String sql;

	// 공급업체 정보를 검색하기 위한 쿼리문을 작성한다.
	sql= “select sid, sname, sceo, saddr, stel from supplier where sid=?”;
	
	//데이터베이스에 연결 한 후 검색할 공급업체 아이디 값을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, sid);
	}

6.6 class Rent

6.6.1 rent()

인터페이스	public void rent(rnum, cid, vnum, rfrom, rto, rpay, rtag)
기능	고객이 비디오를 대여한다.

알고리즘	public void rent(rnum, cid, vnum, rfrom, rto, rpay, rtag)
	{
	// 대여하기 위한 정보를 저장하기 위한 변수를 생성하고 새로 입력되는 값들을 대입한다.
	private String rnum = rnum; //대여 번호
	private String cid = cid; //대여하고자 하는 고객 아이디
	private String vnum = vnum; //비디오 번호
	private String rfrom= rfrom //대여 날짜
	private String rto = rto; //반납 날짜
	private String rpay = rpay; //대여비
	private String rtag = rtag; //대여, 반납 표시 태그
	
	//대여 정보를 데이터베이스에 연결하여 저장하기 위한 url 변수를 생성
	String sql;
	
	//데이터베이스에 공급업체 정보를 추가하기 위한 쿼리문
	sql = “insert into rent(rnum, cid, vnum, rfrom, rto, rpay, rtag) values(?,?,?,?,?,?,?)”;
	
	//데이터베이스와 연결 한 후 추가된 공급업체 정보와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, rnum); //대여번호
	pstmt.setString(2, cid); //대여하고자 하는 고객 번호
	pstmt.setString(3, vnum); //비디오 번호
	pstmt.setString(4, rfrom); //대여 날짜

	pstmt.setString(5, rto); //반납 날짜
	pstmt.setString(6, rpay); //대여비
	pstmt.setString(7, rtag); //대여, 반납 표시 태그
	
	}

6.6.2 overdue()

인터페이스	public void overdue(vnum, cid, vname, overdue, overcharge)
기능	고객의 연체 정보를 등록한다.

알고리즘	public void overdue(vnum, cid, vname, overdue, overcharge)
	{
	// 연체 정보를 저장할 변수를 생성하고 새로 입력되는 값들을 대입한다.
	private String vnum = vnum; //연체된 비디오 번호
	private String cid = cid; //연체 고객 아이디
	private String vname = vname; //연체된 비디오 명
	private String overdue = overdue; //연체한 날짜 수
	private String overcharge = overcharge; //연체료
	
	//연체 정보를 데이터베이스에 연결하여 저장하기 위한 url 변수를 생성
	String sql;
	
	//데이터베이스에 연체 정보를 추가하기 위한 쿼리문
	sql = “insert into overdue(vnum, cid, vname, overdue, overcharge) values(?,?,?,?,?)”;
	
	//데이터베이스와 연결 한 후 추가된 공급업체 정보와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, vnum); //연체한 비디오 번호 db에 추가
	pstmt.setString(2, cid); //연체한 고객 아이디 db에 추가
	pstmt.setString(3, vname); //연체한 비디오 명 db에 추가
	pstmt.setString(4, overdue); //연체 된 날짜 db에 추가
	pstmt.setString(5, overcharge); //연체료 db에 추가
	}

6.6.3 return()

인터페이스	public void return(rid, rtag)
기능	고객이 비디오를 반납하고, 연체와 손상을 검사한다.

알고리즘	public void return(rid, rtag)
	{
	// 반납되었다고 태그를 바꿔주기 위한 변수 생성

```
  private String rtag = rtag;  //대여, 반납태그 수정
  private String rid = rid;  //대여 번호

  //대여 정보를 데이터베이스에 연결하여 저장하기 위한 url 변수를 생성
  String sql;

  //데이터베이스에 반납 태그로 변경하기 위한 쿼리문
  sql = "update rent set rtag=? where=?";

  //데이터베이스와 연결 한 후 변경된 태그 정보와 쿼리문을 보낸다.
  pstmt = conn.prepareStatement(sql);
  pstmt.setString(1, rtag);  //반납이 완료 되었으면 태그를 db에 수정
  pstmt.setString(2, rid);
}
```

6.7 class Order

6.7.1 order()

인터페이스	public void order(onum, vlistnum, sid, sname, count, price, sdate, otag)
기능	공급업체에게 비디오를 주문한다.

알고리즘	public void order(onum, vlistnum, sid, sname, count, price, sdate, otag)
	{
	// 연체 정보를 저장할 변수를 생성하고 새로 입력되는 값들을 대입한다.
	private String onum = onum; //비디오 주문번호
	private String vlistnum = vlistnum; //주문한 비디오 목록 번호
	private String sid = sid; //주문한 공급업체 업체 코드
	private String sname = sname; //주문한 공급업체 명
	private String count = count; //주문한 비디오 수량
	private String price = price; //주문한 비디오 가격
	private String sdate = sdate; //비디오 주문한 날짜
	private String otag = otag; //주문, 납품, 반품 태그
	
	//연체 정보를 데이터베이스에 연결하여 저장하기 위한 url 변수를 생성
	String sql;
	
	//데이터베이스에 공급업체 정보를 추가하기 위한 쿼리문
	sql = “insert into order(onum, vlistnum, sid, sname, count, price, sdate, otag) values(?,?,?,?,?,?,?)”;
	
	//데이터베이스와 연결 한 후 추가된 공급업체 정보와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, onum); //비디오 주문 번호 db에 추가
	pstmt.setString(2, vlistnum); //비디오 목록
	pstmt.setString(3, sid); //공급업체 사원코드
	pstmt.setString(4, sname); //공급업체 이름
	pstmt.setString(5, count); //주문한 비디오 수량
	pstmt.setString(6, price); //주문한 비디오 가격
	pstmt.setString(7, sdate); //주문 날짜
	pstmt.setString(8, otag); //주문, 납품, 반납 태그를 주문값으로 db저장
	}

6.7.2 damagedCheck()

인터페이스	public void damagedCheck(onum, otag, vnum)
기능	주문한 비디오 중 손상된 비디오가 있는지 체크하고 손상되었으면 반품한다.

알고리즘	public void damagedCheck(onum, otag, vnum)
	{
	// 손상된 비디오를 반품 태그로 바꾸기 위한 변수 생성
	private String onum = onum; //주문 번호
	private String otag = otag; //주문상태 태그
	private String vnum = vnum; //비디오 번호
	
	//연체 정보를 데이터베이스에 연결하여 저장하기 위한 url 변수를 생성
	String sql;
	
	//데이터베이스에 공급업체 정보를 추가하기 위한 쿼리문
	sql = “update into order(otag) values(?) where onum=?”;
	
	//데이터베이스와 연결 한 후 추가된 공급업체 정보와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, otag); //주문상태의 태그값을 반품으로 바꾼다.
	pstmt.setString(2, onum);
	}

6.7.3 return()

인터페이스	public void return(onum, otag)
기능	주문한 비디오를 반품한다.

알고리즘	public void return(onum, otag)
	{
	// 비디오를 반품 태그로 바꾸기 위한 변수 생성
	private String onum = onum;
	private String otag = otag; //주문, 납품 상태였던 태그를 반품값으로 바꾼다.
	
	//연체 정보를 데이터베이스에 연결하여 저장하기 위한 url 변수를 생성
	String sql;
	
	//데이터베이스에 공급업체 정보를 추가하기 위한 쿼리문
	sql = “update into order(otag) values(?) where onum=?”;
	
	//데이터베이스와 연결 한 후 추가된 공급업체 정보와 쿼리문을 보낸다.
	pstmt = conn.prepareStatement(sql);
	pstmt.setString(1, otag); //주문상태 혹은 납품 상태였던 태그를 반품값으로 바꾼다.
	pstmt.setString(2, onum);

	}

7. 결론

본 비디오 가게 관리 시스템은 관리자가 비디오 재고 파악, 비디오 주문, 매출 파악, 회원 관리 등을 웹을 통해 편리하게 접근할 수 있다. 사람의 손으로 비디오 매장, 매출, 회원 관리, 대여, 공급 업체 관리 등을 직접 작성하던 사무적인 일들을 모두 컴퓨터가 처리하도록 하여 업무의 신속함과 금전 계산의 정확함, 사용자들의 편의성을 향상시킨다.