

[비디오 가게 관리 시스템]

소프트웨어 공학

<요구사항 분석서>

2018년 11월 04일

소 속 : 충북대학교 소프트웨어학과
이 름 : 구 경 민, 정 예 원
교 수 : 이 종 연 교수님

Contents

1. 서론	4
1.1 개발목표	4
1.2 개발내용	4
1.3 Use Case Diagram	5
1.4 업무분석표	5
2. 데이터 구조 설계	6
2.1 Class-Diagram	6
2.2 스키마 테이블	7
2.3 데이터 구조 설계(자료구조)	10
3. 사용자 인터페이스(UI) 설계	11
4. 시퀀스 다이어그램	14
4.1 Use-Case '대여하다'	14
4.2 Use-Case '반납하다'	15
4.3 Use-Case '주문하다'	14
4.4 Use-Case '반품하다'	14
5. Method 설계	15
5.1 Class Customer	
5.1.1 insert()	
5.1.2 update()	
5.1.3 delete()	
5.1.4 retrieve()	
5.2 Class Employee	
5.2.1 insert()	
5.2.2 update()	
5.2.3 delete()	
5.2.4 retrieve()	
5.3 Class Video	
5.3.1 insert()	

5.3.2	update()	-----
5.3.3	delete()	-----
5.3.4	retrieve()	-----
5.4	Class VideoList	-----
5.4.1	insert()	-----
5.4.2	update()	-----
5.4.3	delete()	-----
5.4.4	retrieve()	-----
5.5	Class Supplier	-----
5.5.1	insert()	-----
5.5.2	update()	-----
5.5.3	delete()	-----
5.5.4	retrieve()	-----
5.6	Class Rent	-----
5.6.1	rent()	-----
5.6.2	overdue()	-----
5.6.3	return()	-----
5.7	Class Order	-----
5.7.1	order()	-----
5.7.2	damagedCheck()	-----
5.7.3	return()	-----
7.	결론	-----

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.

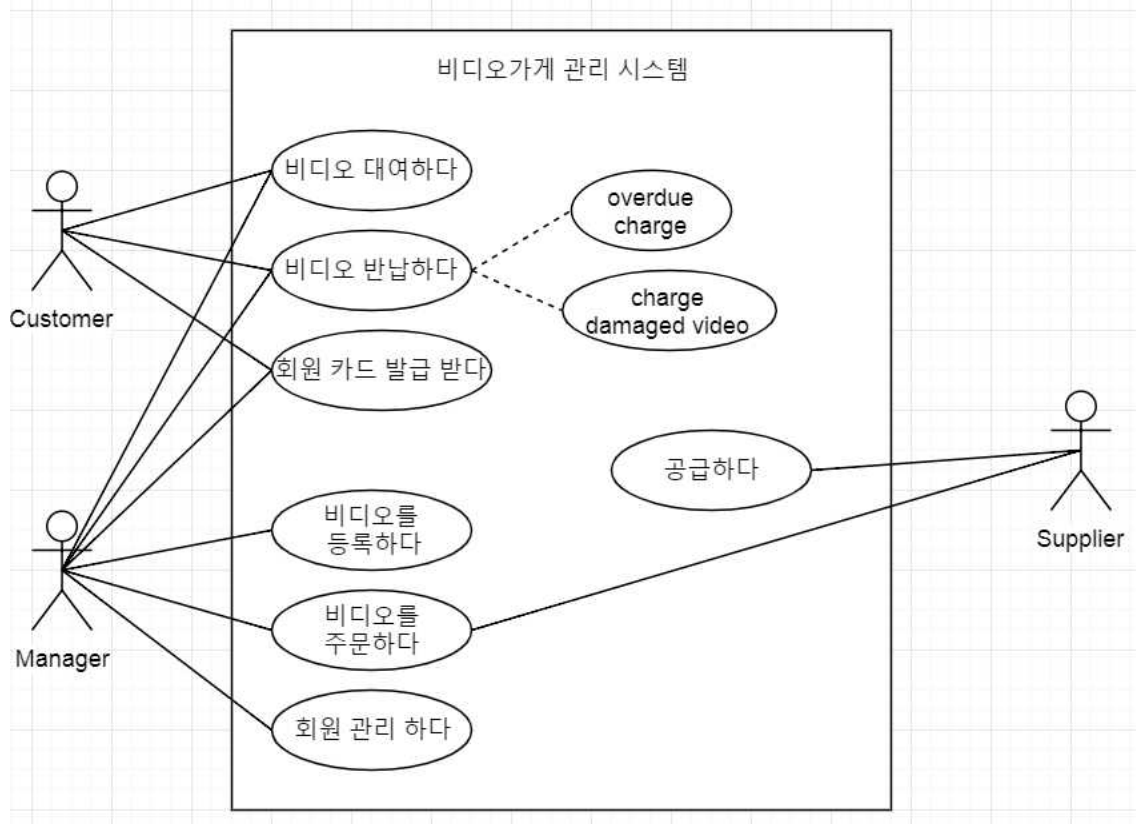
1.1 개발 목표

- 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

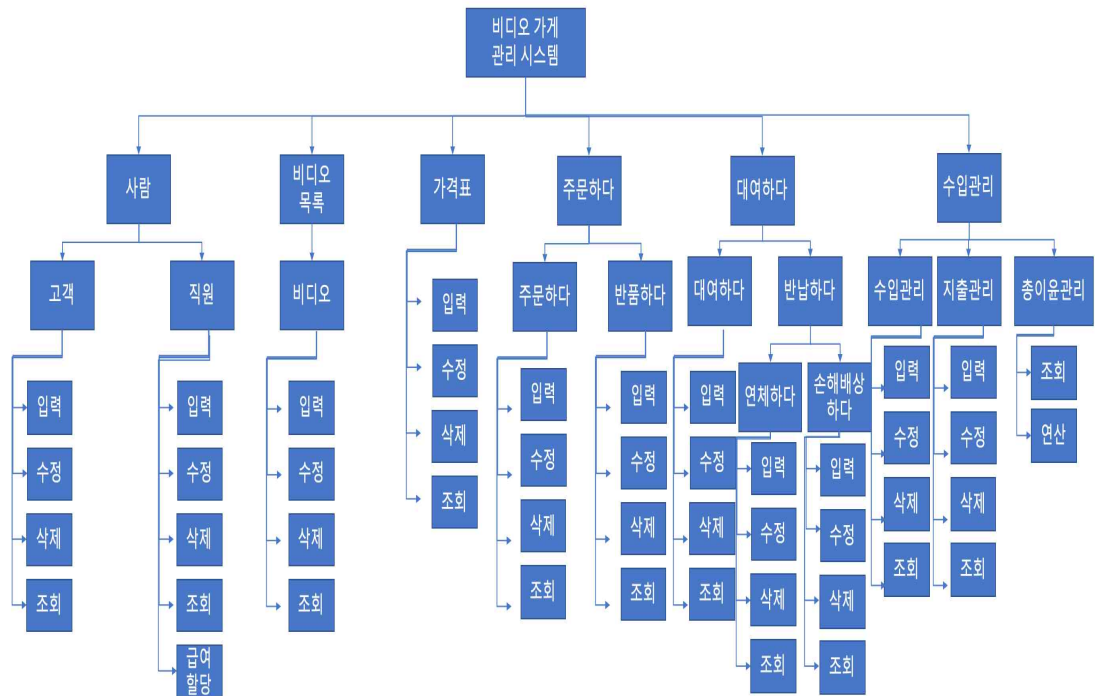
1.2 개발 내용

분류	개발 내용
소비자	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.
공급업체	1) 공급업체 관리: 서버 관리자로부터 공급업체들을 관리한다. 2) 업체별 공급제품관리: 업체별로 공급하는 비디오가 다르기 때문에 나누어서 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

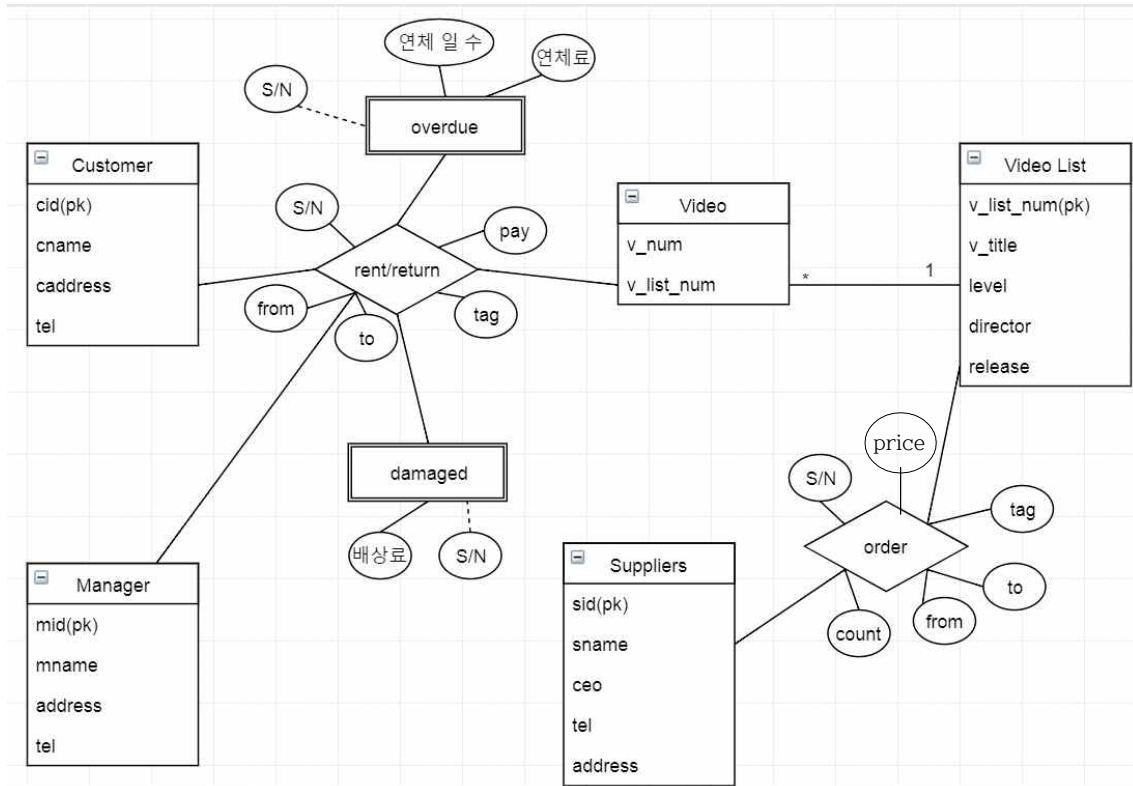


1.4 업무분석표



2. 데이터 구조 설계

2.1 Class-Diagram



2.2 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	NNs

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

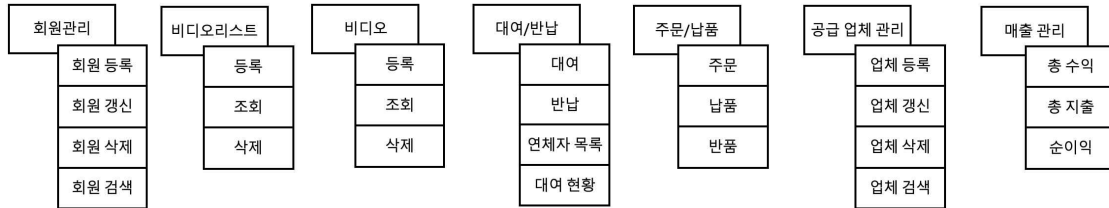
공급업체(Supplier)				
업체번호	sid	INT	10	PK
업체명	sname	CHAR	50	NN
업체ceo	sceo	CHAR	50	NN
업체주소	saddress	CHAR	100	NN
전화번호	stel	CHAR	20	NN

2.3 데이터 구조 설계(자료구조)

클래스	내용
Costomer	c_id : int c_name : string c_address : string c_tel : string c_eamil : string
Manager	m_id : int m_name : string m_address : string m_tel : string m_email : string m_salary : string m_position : string
Supplier	S_ID :int S_Number :int S_CEO : string S_address : string S_tel : string
Video	v_num : int v_list_num : int v_name : string v_director : string v_release : string v_supplier : string v_price : string v_amount : int

3. 사용자 인터페이스(UI) 설계

3.1 메인 메뉴



3.2 회원 관리 메뉴

회원관리

회원등록

이름

주소

전화번호

정보를 입력하세요.

확인

회원관리

회원 갱신

이름

주소

전화번호

수정할 정보를 입력하세요.

확인

회원관리

회원 조회

아이디

조회할 아이디를 입력하세요.

확인

회원관리

회원 삭제

아이디

회원을 삭제하시겠습니까?

확인

3.3 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

3.4 비디오 메뉴

비디오

등록

비디오 번호

비디오 리스트 번호

등록할 정보를 입력하세요.

확인

비디오

삭제

비디오 번호

삭제할 번호를 입력하세요.

확인

비디오

조회

비디오 번호

조회할 번호를 입력하세요.

확인

3.5 대여/반납 메뉴

대여

고객 번호

비디오 번호

대여 일

반납 일

가격

확인

반납

대여 번호

확인

연체자 목록 조회

고객 번호

확인

대여 현황

대여 번호

확인

3.6 주문/반품 메뉴

주문

주문 번호

비디오 번호

공급 업체

수량

날짜

확인

반품

주문 번호

비디오 번호

수량

from

to

확인

납품

주문 번호

입력한 주문이 납품 되었습니까?

확인

3.7 공급 업체 관리 메뉴

공급업체 관리

등록

업체 번호	<input type="text"/>
업체 명	<input type="text"/>
CEO	<input type="text"/>
주소	<input type="text"/>
전화번호	<input type="text"/>

등록할 업체 정보를 입력하세요.

확인

공급업체 관리

갱신

업체 번호	<input type="text"/>
업체 명	<input type="text"/>
CEO	<input type="text"/>
주소	<input type="text"/>
전화번호	<input type="text"/>

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

3.8 매출 관리

매출 관리

총 수익

총 수익은

입니다.

확인

매출 관리

총 지출

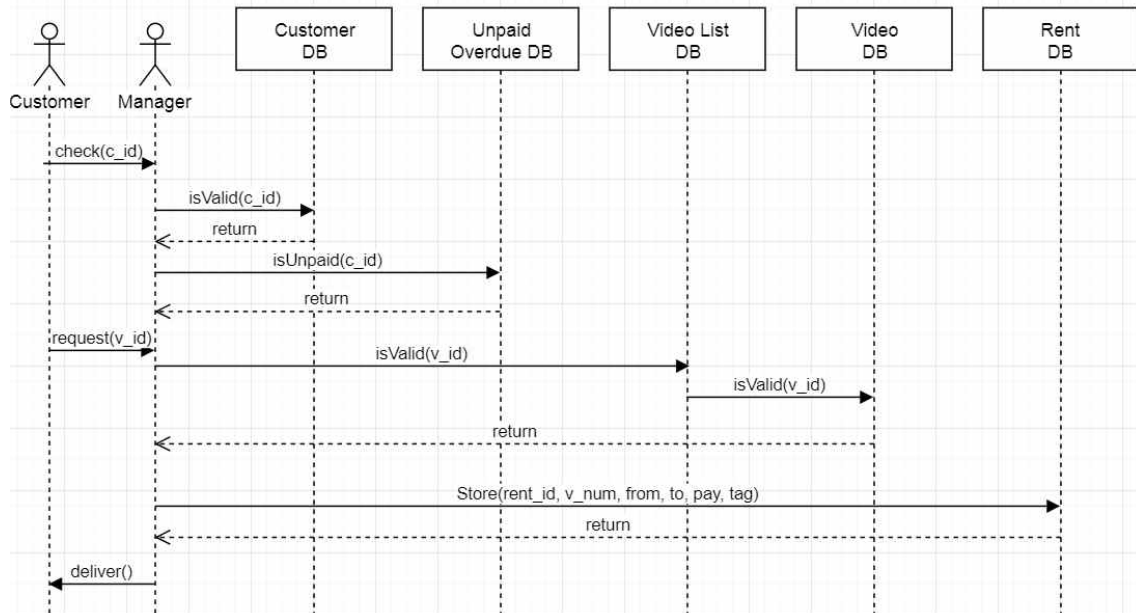
총 지출은

입니다.

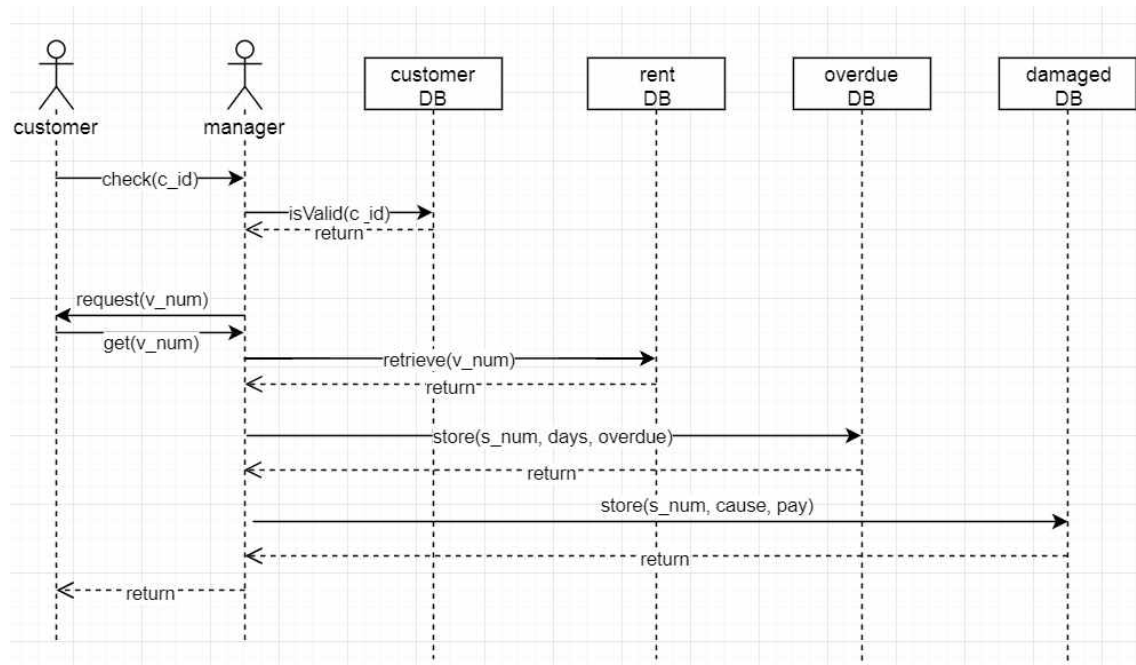
확인

4. 시퀀스 다이어그램

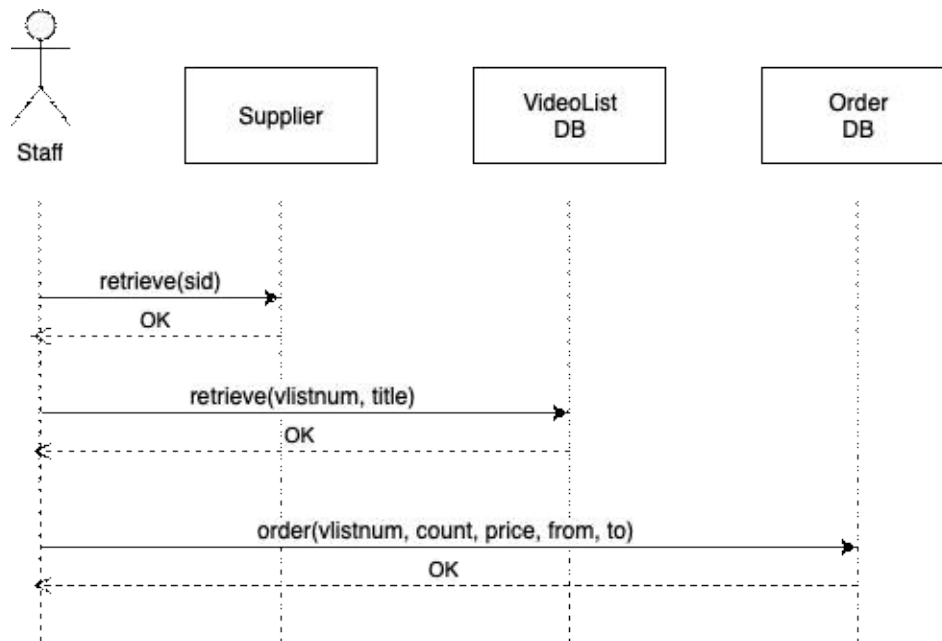
4.1 Use-Case '대여하다'



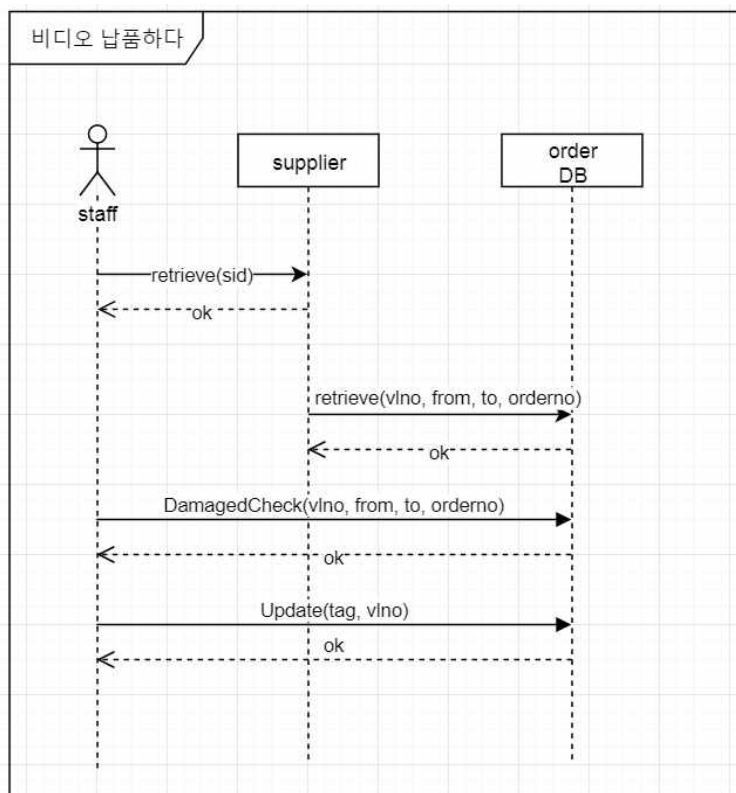
4.2 Use-Case '반납하다'



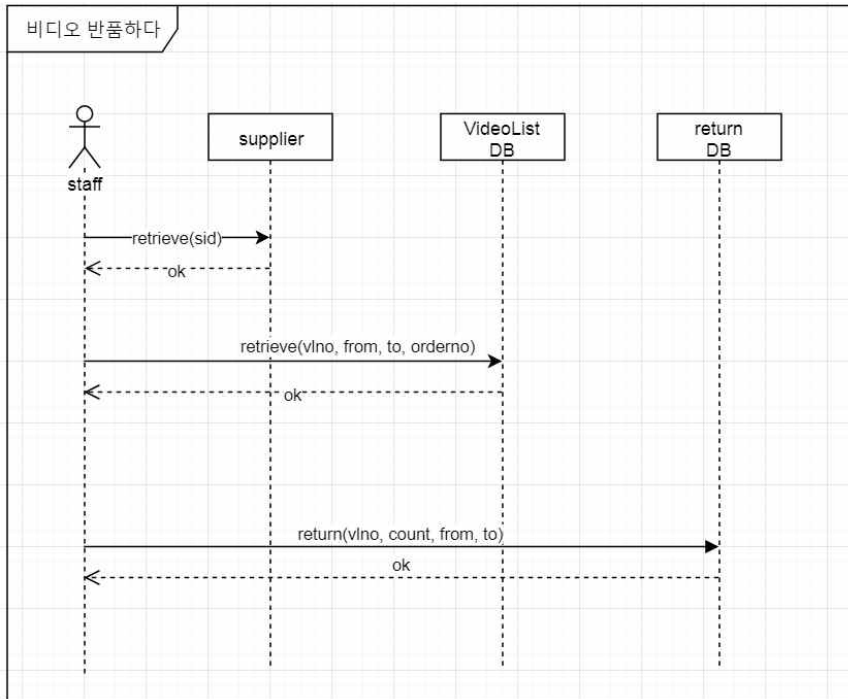
4.3 Use-Case '주문하다'



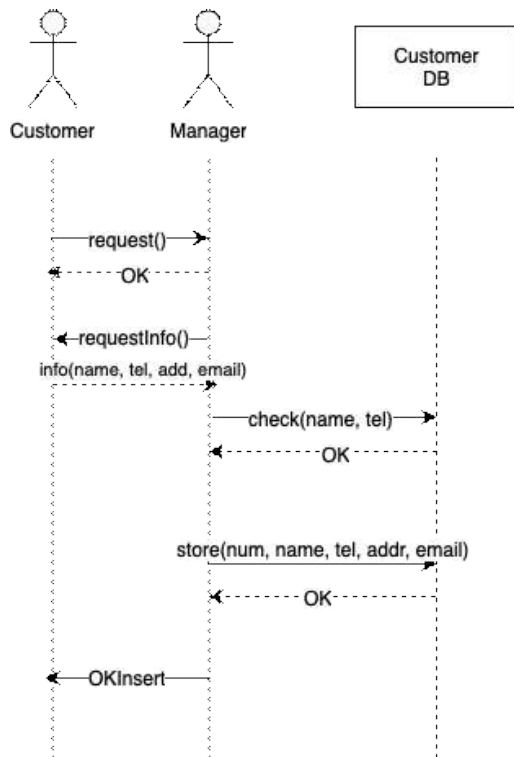
4.4 Use-Case '납품하다'



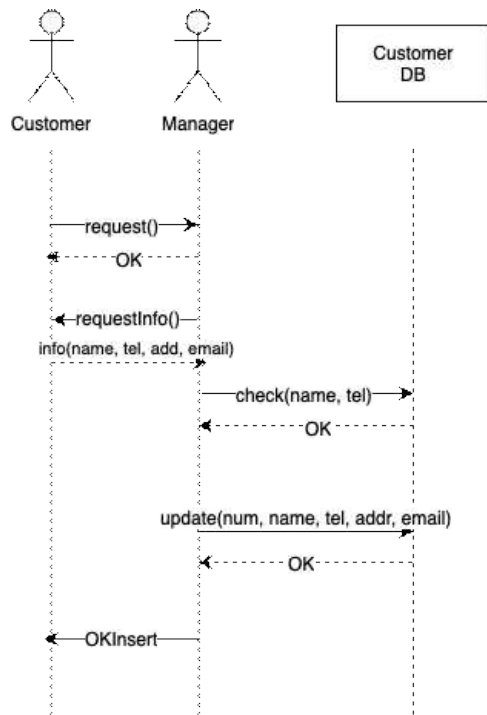
4.5 Use-Case '반품하다'



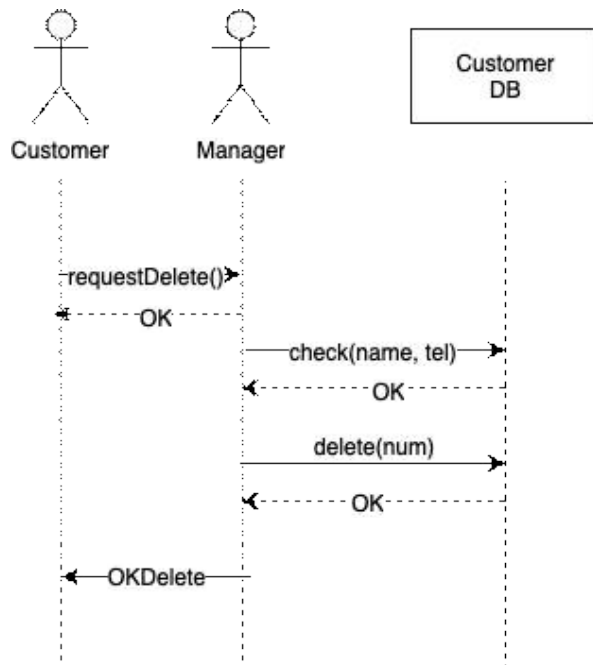
4.4 Use-Case '회원 등록하다'



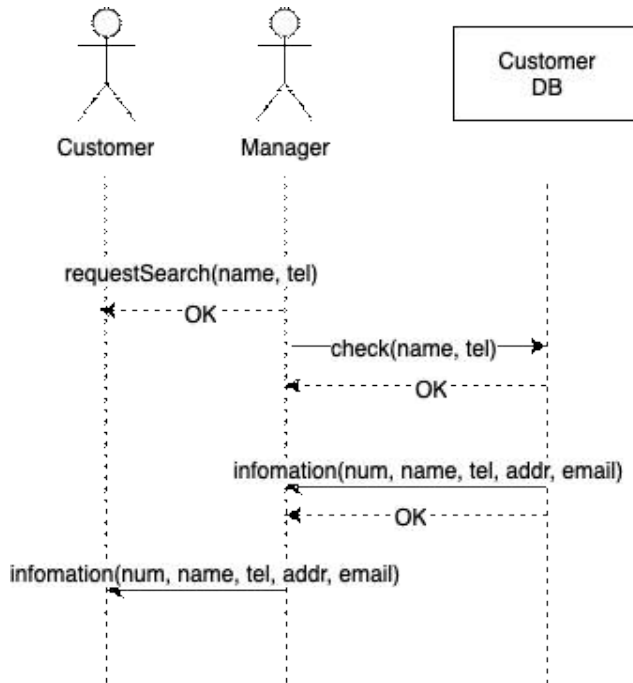
4.5 Use-Case '회원 갱신하다'



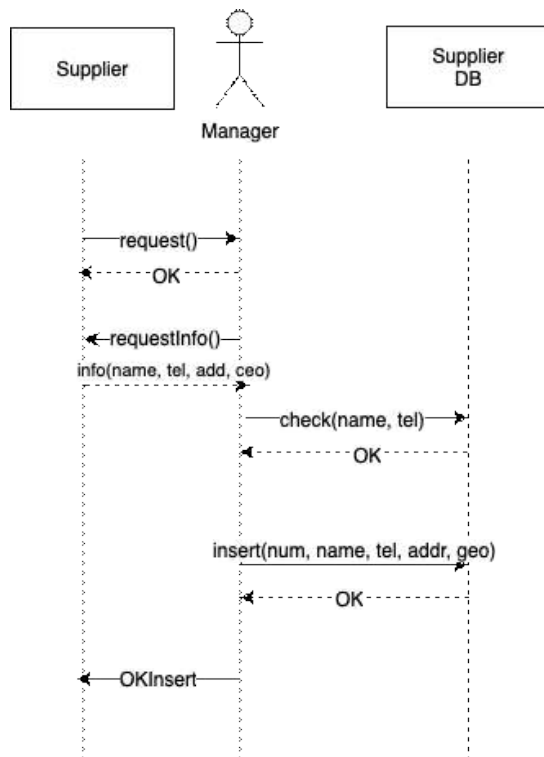
4.6 Use-Case '회원 삭제하다'



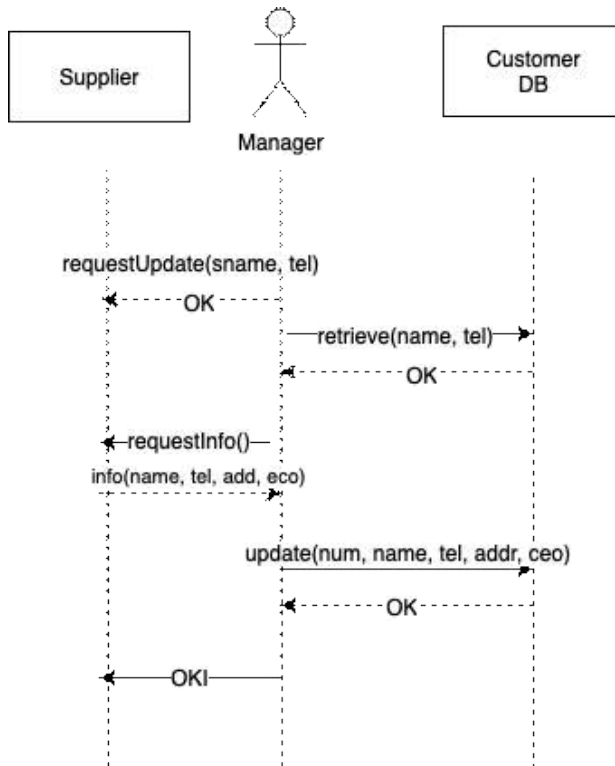
4.7 Use_Case '회원 조회하다'



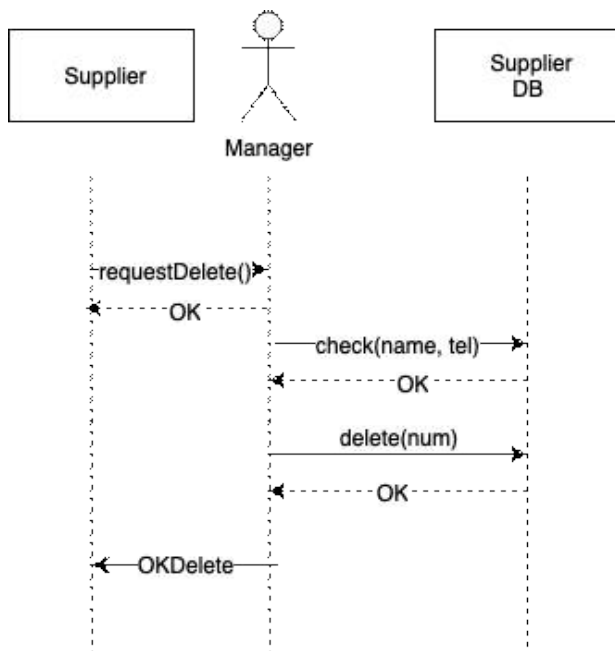
4.8 Use-Case '공급업체 등록하다'



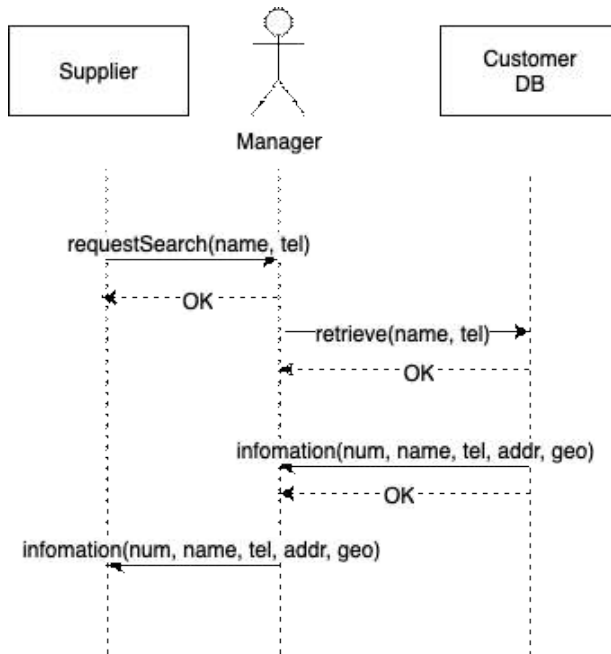
4.9 Use-Case '공급업체 갱신하다'



4.10 Use-Case '공급업체 삭제하다'



5.11 Use-Case '공급업체 조회하다'



5. Method 설계

5.1 class Customer

5.1.1 insert()

인터페이스	public void customerInsert(cid, cname, caddress, ctel)
기능	고객 데이터베이스에 고객 정보를 저장한다.
알고리즘	<pre> public void customerInsert(cid, cname, caddress, ctel) { //고객 정보에 저장할 변수를 생성하고 새로 입력되는 값들을 대입한다. private String cid=cid; private String cname=cname; private String caddress=caddress; private String ctel=ctel; //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); pstmt.setString(2, cname); pstmt.setString(3, caddress); pstmt.setString(4, ctel); } </pre>

5.1.2 update()

인터페이스	public void customerUpdate(cid, cname, caddress, ctel)
기능	고객 데이터베이스에 저장된 고객 정보를 수정한다.
알고리즘	<pre> public void customerUpdate(cid, cname, caddress, ctel) { //수정할 고객정보의 아이디, 이름, 주소, 전화번호에 해당하는 변수를 생 성한다. private String cid=cid; private String cname=cname; private String caddress=caddress; } </pre>

	<pre> private String ctel=ctel; //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); pstmt.setString(2, cname); pstmt.setString(3, caddress); pstmt.setString(4, ctel); } </pre>
--	---

5.1.3 delete()

인터페이스	public void customerDelete(cid)
기능	고객 데이터베이스에 저장된 고객 정보를 삭제한다.
알고리즘	<pre> public void customerDelete(cid) { //삭제할 고객의 아이디를 저장할 변수를 생성한다. String cid=cid; //데이터베이스에 쿼리명령을 보내기 위한 sql 변수를 생성한다. String 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); } </pre>

5.1.4 retrieve()

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

	<pre>//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); }</pre>
--	---

5.2 class Employee

5.2.1 insert()

인터페이스	public void empInsert(mid, mname, maddress, mtel)
기능	직원 데이터베이스에 직원 정보를 저장한다.
알고리즘	<pre> public void empInsert(mid, mname, maddress, mtel) { //직원 정보에 저장할 변수를 생성하고 새로 입력되는 값들을 대입한다. private String mid=mid; private String mname=mname; private String maddress=maddress; private String mtel=mtel; //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); pstmt.setString(2, mname); pstmt.setString(3, maddress); pstmt.setString(4, mtel); } </pre>

5.2.2 update()

인터페이스	public void empUpdate(mid, mname, maddress, mtel)
기능	직원 데이터베이스에 저장된 직원 정보를 수정한다.
알고리즘	<pre> public void empUpdate(mid, mname, maddress, mtel) { //수정할 직원 정보의 아이디, 이름, 주소, 전화번호에 해당하는 변수를 생 성한다. private String mid=mid; private String mname=mname; private String maddress=maddress; private String mtel=mtel; } </pre>

	<pre>//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); pstmt.setString(2, mname); pstmt.setString(3, maddress); pstmt.setString(4, mtel); }</pre>
--	---

5.2.3 delete()

인터페이스	public void empDelete(mid)
기능	직원 데이터베이스에 저장된 직원 정보를 삭제한다.
알고리즘	<pre>public void empDelete(mid) { //삭제할 직원의 아이디를 저장할 변수를 생성한다. String mid=mid; //데이터베이스에 쿼리명령을 보내기 위한 sql 변수를 생성한다. String 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); }</pre>

5.2.4 retrieve()

인터페이스	public void empRetrieve(mid)
기능	직원 데이터베이스에 저장된 직원 정보를 검색한다.
알고리즘	<pre>public void empRetrieve(mid) { //검색할 고객의 아이디를 저장할 변수를 생성한 후 대입한다. String mid=mid;</pre>

	<pre>//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); }</pre>
--	---

5.3 class Video

5.3.1 insert()

인터페이스	public void videoInsert(vnum, vlistnum)
기능	비디오 데이터베이스에 비디오 정보를 저장한다.
알고리즘	<pre> public void videoInsert(vnum, vlistnum) { //비디오 정보에 저장할 변수를 생성하고 새로 입력되는 값들을 대입한다. private String vnum = vnum; private String vlistnum = vlistnum; //비디오 정보를 데이터베이스에 연결하여 저장하기 위한 url 변수를 생성 String sql; //데이터베이스에 비디오 정보를 추가하기 위한 쿼리문 sql = "insert into video(vnum, vlistnum) values(?,?)"; //데이터베이스와 연결 한 후 추가된 비디오 정보와 쿼리문을 보낸다. pstmt = conn.prepareStatement(sql); pstmt.setString(1, vnum); pstmt.setString(2, vlistnum); } </pre>

5.3.2 update()

인터페이스	public void videoUpdate(vnum, vlistnum)
기능	비디오 데이터베이스에 저장된 비디오를 수정한다.
알고리즘	<pre> public void videoUpdate(vnum, vlistnum) { //수정할 비디오 정보의 비디오번호, 비디오리스트번호에 해당하는 변수를 생성한다. private String vnum = vnum; private String vlistnum = vlistnum; //데이터베이스에 연결할 쿼리문을 작성하기 위한 변수를 생성한다. String sql; //비디오 정보 업데이트를 하기 위한 쿼리문을 작성한다. sql= "update video set vnum=?, vlistnum=?"; //데이터베이스와 연결 한 후 수정된 비디오 정보와 쿼리문을 보낸다. } </pre>

	<pre> pstmt = conn.prepareStatement(sql); pstmt.setString(1, vnum); pstmt.setString(2, vlistnum); } </pre>
--	--

5.3.3 delete()

인터페이스	public void videoDelete(vnum, vlistnum)
기능	비디오 데이터베이스에 저장된 비디오를 삭제한다.
알고리즘	<pre> public void videoDelete(vnum, vlistnum) { //삭제할 비디오번호를 저장할 변수를 생성한다. String vnum = vnum; //데이터베이스에 쿼리명령을 보내기 위한 sql 변수를 생성한다. String 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); } </pre>

5.3.4 retrieve()

인터페이스	public void videoRetrieve(vnum)
기능	비디오 데이터베이스에 저장된 비디오 정보를 검색한다.
알고리즘	<pre> 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); } </pre>

5.4 class VideoList

5.4.1 insert()

인터페이스	public void videoListInsert(vlistnum, vname, vrelease, vprice, vamount, vdirector)
기능	비디오 리스트 데이터베이스에 비디오 리스트 정보를 저장한다.
알고리즘	<pre> public void videoListInsert(vlistnum, vname, vrelease, vprice, vamount, vdirector) { //비디오 리스트 정보에 저장할 변수를 생성하고 새로 입력되는 값들을 대입한다. private String vlistnum = vlistnum; private String vname = vname; private String vrelease = vrelease; private String vprice = vprice; private String vamount = vamount; private String vdirector = vdirector; //비디오 리스트 정보를 데이터베이스에 연결하여 저장하기 위한 url 변수 를 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); pstmt.setString(2, vname); pstmt.setString(3, vrelease); pstmt.setString(4, vprice); pstmt.setString(5, vamount); pstmt.setString(6, vdirector); } </pre>

5.4.2 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,

	<pre> vamount, vdirector) { //수정할 비디오 리스트 정보의 비디오리스트번호, 비디오이름, 개봉일, 가격, 수량, 감독에 해당하는 변수를 생성한다. private String vlistnum = vlistnum; private String vname = vname; private String vrelease = vrelease; private String vprice = vprice; private String vamount = vamount; private String vdirector = vdirector; //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); pstmt.setString(2, vname); pstmt.setString(3, vrelease); pstmt.setString(4, vprice); pstmt.setString(5, vamount); pstmt.setString(6, vdirector); } </pre>
--	--

5.4.3 delete()

인터페이스	public void videoListDelete(vlistnum)
기능	비디오 리스트 데이터베이스에 비디오 리스트 정보를 삭제한다.
알고리즘	<pre> public void videoListDelete(vlistnum) { //삭제할 비디오리스트번호를 저장할 변수를 생성한다. String vlistnum = vlistnum; //데이터베이스에 쿼리명령을 보내기 위한 sql 변수를 생성한다. String sql; //해당 비디오 리스트 아이디를 데이터베이스에서 삭제하기 위한 쿼리문을 </pre>

	<p>작성한다.</p> <pre>sql = "delete from videolist where vlistnum=?";</pre> <p>//데이터베이스와 연결 한 후 삭제할 비디오리스트번호와 쿼리문을 보낸다.</p> <pre>pstmt = conn.prepareStatement(sql); pstmt.setString(1, vlistnum); }</pre>
--	--

5.4.4 retrieve()

인터페이스	public void videoListRetrieve(vlistnum)
기능	비디오 리스트 데이터베이스에 비디오 리스트 정보를 검색한다.
알고리즘	<pre>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); }</pre>

5.5 class Supplier

5.5.1 insert()

인터페이스	public void supplierInsert(sid, sname, sceo, saddr, stel)
기능	공급업체 데이터베이스에 공급업체 정보를 저장한다.
알고리즘	<pre>public void supplierInsert(sid, sname, sceo, saddr, stel) { //공급업체 정보에 저장할 변수를 생성하고 새로 입력되는 값들을 대입한 다. private String sid = sid; // 공급업체 번호 private String sname = sname; // 공급업체 명 private String sceo = sceo; // 공급업체 ceo</pre>

	<pre> 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); pstmt.setString(2, sname); pstmt.setString(3, sceo); pstmt.setString(4, saddr); pstmt.setString(5, stel); } </pre>
--	---

5.5.2 update()

인터페이스	public void supplierUpdate(sid, sname, sceo, saddr, stel)
기능	공급업체 데이터베이스에 저장된 공급업체 정보를 수정한다.
알고리즘	<pre> public void supplierUpdate(sid, sname, sceo, saddr, stel) { //수정할 공급업체 정보의 아이디, 이름, ceo, 주소, 전화번호에 해당하는 변수를 생성한다. private String sid = sid; // 공급업체 번호 private String sname = sname; // 공급업체 이름 private String sceo = sceo; // 공급업체 ceo 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. </pre>

	<pre> pstmt = conn.prepareStatement(sql); pstmt.setString(1, sid); pstmt.setString(2, sname); pstmt.setString(3, sceo); pstmt.setString(4, saddr); pstmt.setString(5, stel); } </pre>
--	---

5.5.3 delete()

인터페이스	public void supplierDelete(sid)
기능	공급업체 데이터베이스에 저장된 공급업체 정보를 삭제한다.
알고리즘	<pre> public void supplierDelete(sid) { //삭제할 공급업체 아이디를 저장할 변수를 생성한다. String sid = sid; // 공급업체 번호 //데이터베이스에 쿼리 명령을 보내기 위한 sql 변수를 생성한다. String sql; //해당 공급업체 아이디를 데이터베이스에서 삭제하기 위한 쿼리문을 작성 한다. sql = "delete from supplier where sid=?"; //데이터베이스와 연결 한 후 삭제할 공급업체아이디와 쿼리문을 보낸다. pstmt = conn.prepareStatement(sql); pstmt.setString(1, sid); } </pre>

5.5.4 retrieve()

인터페이스	public void supplierRetrieve(sid)
기능	공급업체 데이터베이스에 저장된 공급업체 정보를 검색한다.
알고리즘	<pre> public void supplierRetrieve(sid) { //검색할 공급업체 아이디를 저장할 변수를 생성한 후 대입한다. String sid = sid; // 공급업체 번호 //데이터베이스에 보낼 쿼리문을 담을 변수를 생성한다. String sql; </pre>

	<pre>// 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); }</pre>
--	--

5.6 class Rent

5.6.1 rent()

인터페이스	public void rent(rnum, cid, vnum, rfrom, rto, rpay, rtag)
기능	고객이 비디오를 대여한다.
알고리즘	<pre>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; // 고객 번호 private String vnum = vnum; // 비디오 번호 private String rfrom= rfrom // 대여날짜 private String rto = rto; // 반납날짜 private String rpay = rpay; // 대여료 private String rtag = rtag; // 주문반납 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); pstmt.setString(2, cid); pstmt.setString(3, vnum); pstmt.setString(4, rfrom); pstmt.setString(5, rto);</pre>

	<pre> pstmt.setString(6, rpay); pstmt.setString(7, rtag); } </pre>
--	---

5.6.2 overdue()

인터페이스	public void overdue(vnum, cid, vname, overdue, overcharge)
기능	고객의 연체 정보를 등록한다.
알고리즘	<pre> public void overdue(vnum, cid, vname, overdue, overcharge) { // 연체 정보를 저장할 변수를 생성하고 새로 입력되는 값들을 대입한다. private String vnum = vnum; // 비디오 번호 private String cid = cid; // 고객 번호 private String vname = vname; // 비디오 이름 private String overdue = overdue; // 연체일 private String overcharge = overcharge; // 연체료 //연체 정보를 데이터베이스에 연결하여 저장하기 위한 url 변수를 생성 String sql; //데이터베이스에 공급업체 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); pstmt.setString(2, cid); pstmt.setString(3, vname); pstmt.setString(4, overdue); pstmt.setString(5, overcharge); } </pre>

5.6.3 return()

인터페이스	public void return(rid, rtag)
기능	고객이 비디오를 반납하고, 연체와 손상을 검사한다.
알고리즘	<pre> public void return(rid, rtag) { // 반납되었다고 태그를 바꿔주기 위한 변수 생성 private String rid = rid; // 대여 번호 </pre>

	<pre>private String rtag = rtag; // 대여연체 태그 //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); pstmt.setString(2, rid); }</pre>
--	---

5.7 class Order

5.7.1 order()

인터페이스	public void order(onum, vlistnum, sid, sname, count, price, sdate, otag)
기능	공급업체에게 비디오를 주문한다.
알고리즘	<pre> public void order(onum, vlistnum, sid, sname, count, price, sdate, otag) { // 연체 정보를 저장할 변수를 생성하고 새로 입력되는 값들을 대입한다. private String onum = onum; // 주문 번호 private String vlistnum = vlistnum; // 비디오리스트 번호 private String sid = sid; // 공급업체 번호 private String sname = sname; // 공급업체 명 private String count = count; // 수량 private String price = price; // 단가 private String sdate = sdate; // 배송 일자 private String otag = otag; // 주문반품 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); pstmt.setString(2, vlistnum); pstmt.setString(3, sid); pstmt.setString(4, sname); pstmt.setString(5, count); pstmt.setString(6, price); pstmt.setString(7, sdate); pstmt.setString(8, otag); } </pre>

5.7.2 damagedCheck()

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

알고리즘	<pre> public void damagedCheck(onum, otag, vnum) { // 손상된 비디오를 반품 태그로 바꾸기 위한 변수 생성 private String onum = onum; // 주문 번호 private String otag = otag; // 주문반품 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); pstmt.setString(2, onum); } </pre>
------	--

5.7.3 return()

인터페이스	public void return(onum, otag)
기능	주문한 비디오를 반품한다.
알고리즘	<pre> public void return(onum, otag) { // 비디오를 반품 태그로 바꾸기 위한 변수 생성 private String onum = onum; // 주문번호 private String otag = otag; // 주문반품 태그 //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); pstmt.setString(2, onum); } </pre>

6. 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.