

# 이상거래 탐지 시스템

## FDS

---

김민희 정민교 정유철 채희성

# 목차

table of contents

1	기획의도 및 배경	9	물리설계
2	목표	10	시스템 아키텍처
3	기대효과	11	기능명세서
4	제안의 특징점	12	스토리보드
5	주요기능	13	<u>소스코드</u>
6	개발방법론	14	일정(WBS)
7	마인드맵	15	조직도(WBS)
8	FDS 요구사항	16	사용된 프로그램

## 01. 기획의도 및 배경

### PG 결제시스템의 중요성

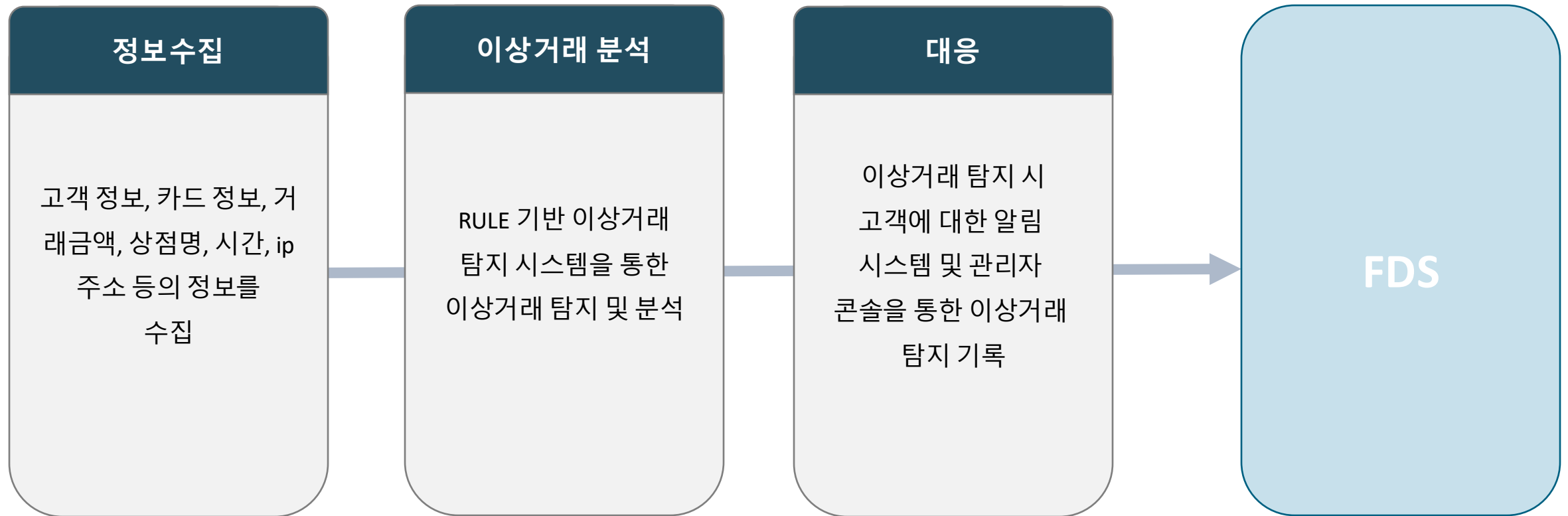
PG사는 전자상거래와 온라인 결제의 핵심 역할을 합니다. 안전하고 신속한 결제 시스템은 고객 경험에 있어 아주 중요한 요소이며 이상거래는 금융적 손실 뿐만 아니라 이러한 고객 경험과 고객의 신뢰를 훼손시킬 수 있습니다. 때문에 실시간으로 이상거래를 탐지하고 조치를 취함으로써 고객 신뢰 유지와 금융적 안정성 확보에 도움이 됩니다.

### 신용카드 거래

신용카드 거래는 소비자들 사이에서 가장 널리 사용되는 결제 방식이며 신용카드 정보가 유출·도용될 경우, 고객과 기업 모두에게 큰 문제가 될 수 있습니다. 이러한 신용카드 사기를 예방하고 고객 신뢰를 유지하기 위해 FDS를 구축하고 강화해야 할 필요성이 있습니다.

### 상품권 거래

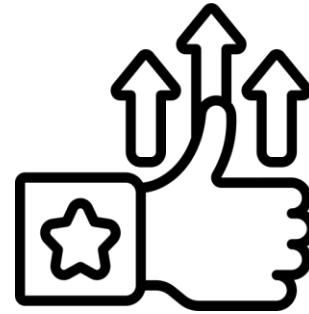
상품권 거래는 실제 결제 시 현금과 동일하게 사용되며, 그 특성 상 환불과 추적이 어려워 사기 행위에 이용될 수 있는 가능성이 매우 높습니다. 이러한 상품권 거래의 특수한 거래방식에 대응하기 위해 정확한 모니터링과 이상거래 패턴 인식에 의한 FDS 구축이 필수적입니다.





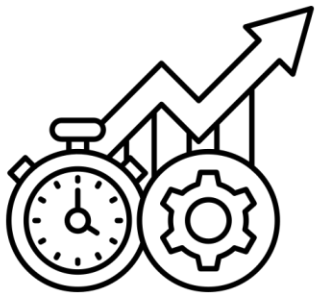
#### ▣FDS를 통한 신속한 탐지와 대응

- FDS를 통해 이상거래를 신속하게 탐지하고 분석할 수 있어 사기 행위 발생 시 즉각적인 대응이 가능합니다.



#### ▣고객 신뢰 증대

- 안전한 결제 환경을 제공함으로써, 고객들의 신뢰를 증대시키고 장기적인 고객 충성도를 유지할 수 있으며 이는 회사의 브랜드 이미지와 평판에도 긍정적인 영향을 미칩니다.



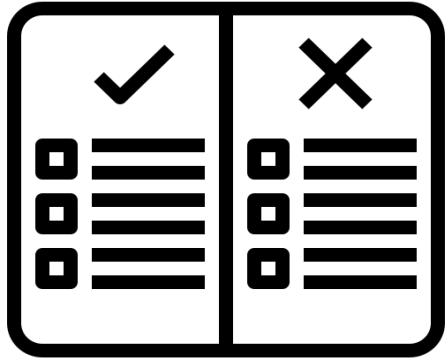
#### ▣운영 효율성 향상

- 이상거래 탐지 시스템을 도입함으로써, 효율성을 크게 향상시킬 수 있습니다. 업무 프로세스의 단순화와 시간 절감에 기여하여 전반적인 운영 비용을 줄일 수 있습니다.



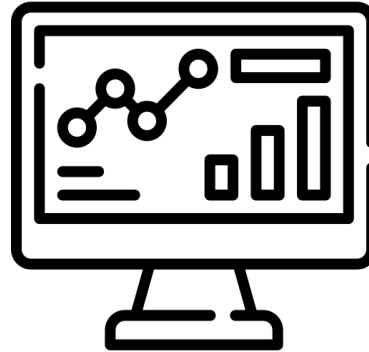
#### ▣시장 경쟁력 강화

- 고객 데이터 보호와 사기 방지에 대한 강력한 대응 능력을 통해 시장에서의 경쟁력을 높일 수 있으며, 안정성과 신뢰성의 강조를 통해 더 많은 비즈니스 기회를 확보할 수 있습니다.



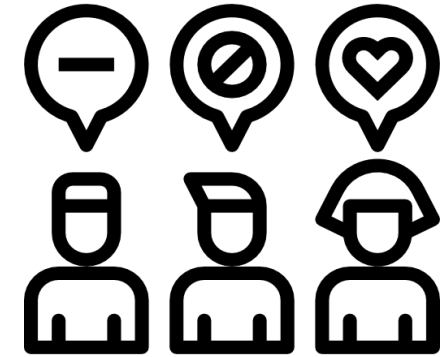
## 다수의 RULE

상세하게 설정된 다수의 RULE을 통해 특정 유형의 이상거래에 대해 신속하고 정확하게 탐지할 수 있습니다.



## 실시간 모니터링

이상거래 발생 시, 고객에게 즉시 알림이 발생하며 관리자는 관리자 콘솔을 통해 거래 내역을 포함한 모든 데이터들을 실시간으로 모니터링이 가능하며 다수의 실시간 차트를 통해 한 눈에 보기 쉽게 데이터를 확인할 수 있습니다.



## 사용자 정의 가능성

기존 RULE 뿐만 아니라 각 고객의 특수한 요구에 맞춘 새로운 RULE 설정을 추가하여 확장성을 높일 수 있습니다.

## 05. 주요기능

RULE 기반의 이상거래 탐지 시스템

Chart.js를 이용한 실시간 데이터의 시각화

ajax를 이용한 비동기적 페이지 전환



Chart.js

## 06. 개발방법론

### Phase 1 구축 시스템 청사진 제시

- ✚ 현행 시스템 지원 기능 분석
- ✚ 시스템에 대한 요구 사항 분석
- ✚ 시스템 요구 사항에 대한 우선 순위 설정
- ✚ 시스템 청사진 제시
- ✚ 개발 범위 확정
- ✚ 개발 범위에 대한 검토 및 확인
- ✚ 시스템의 기본요건 도출
- ✚ 우선 순위에 기반 하여 개발 순위 확정
- ✚ 개발계획의 상세화

### Phase 2 목표 업무 분석

- ✚ 목표기능 현상평가
- ✚ 목표기능 요구사항 분석 및 정의
- ✚ 개발 환경구성
- ✚ 개발목표 업무기능의 상세요구사항 분석 및 정의
- ✚ 요구사항대비 현상의 Gap 분석
- ✚ 개발범위의 상세 합의 도출
- ✚ 목표기능 구성도
- ✚ 목표 데이터 분석서

### Phase 3 목표 업무 설계

- ✚ 목표기능 모델링
- ✚ 목표기능 기능설계
- ✚ 프로세스, 데이터, 이벤트, 분산을 기점으로 기본 설계, 상세 설계, Spec 작성
- ✚ Process Map
- ✚ DFD, Event List
- ✚ ERD, DB schema
- ✚ 분산 구성도

### phase 4 목표 업무 개발

- ✚ A/P 코딩
- ✚ 단위,통합,인수 테스트
- ✚ 시스템 튜닝
- ✚ 데이터전환
- ✚ 매뉴얼작성
- ✚ 단위 테스트 병행
- ✚ 통합 테스트, 시스템 테스트, 성능 테스트
- ✚ 튜닝 병행, 인수
- ✚ 테스트 결과 확인
- ✚ 매뉴얼
- ✚ 테스트 결과 보고서

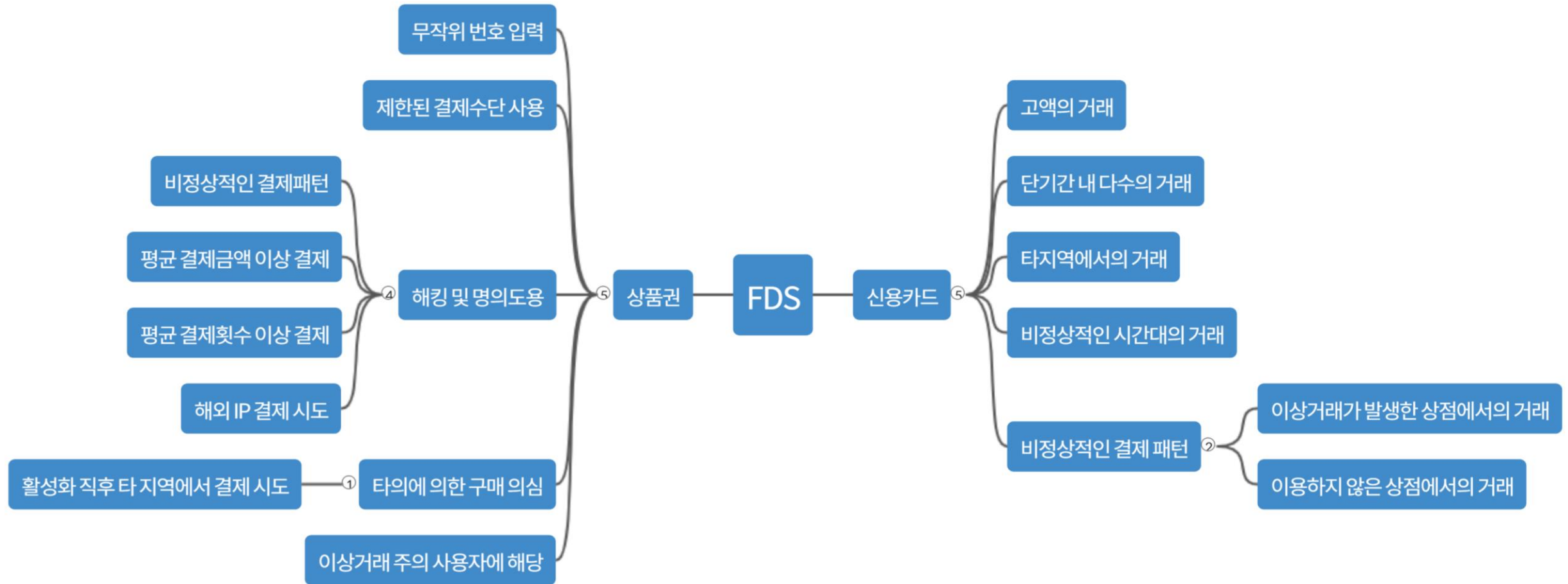
### Phase 5 목표 업무 이행

- ✚ 운용자 교육
- ✚ 시범 운영
- ✚ DRP검토
- ✚ 운영
- ✚ 개발 완료된 시스템
- ✚ 교육 실시 및 시범 운영 실시
- ✚ 실행에 따른 DRP 검토
- ✚ 완료 보고서

품질 관리 계획 수립

품질 보증 활동 → 주기적인 품질 Review Meeting, 변경 관리, 표준 관리 등





## 08. FDS요구사항

신용카드 FDS		
번호	요인	해결
1	특정 금액 이상의 거래. 단일 거래가 일정 금액을 초과하는 경우 이를 이상거래로 간주할 수 있다.	일정 금액(50만원) 이상의 거래가 발생했을 때, 해당 카드로 이전에 50만원 이상의 금액을 결제한 기록이 없으면 이상으로 탐지한다.
2	일정 기간 동안의 여러 거래. 단일 고객이 짧은 시간 동안에 여러 거래를 수행하는 경우 이를 이상거래로 간주할 수 있다.	해당 카드의 가장 최근 결제 시간을 그 다음으로 최근인 결제의 결제 시간 및 현재 시간과 각각 비교해서 그 간격이 5분 이내일 경우 이상으로 탐지한다.
3	타지역에서의 거래. 고객이 일반적으로 활동하는 지역과 다른 지역에서 거래가 발생할 경우 이를 이상거래로 간주할 수 있다.	가장 최근 결제했을 때의 ip 주소가 현재 ip 주소와 다를 경우 이상으로 탐지한다.
4	비정상적인 시간대의 거래. 일반적인 거래 시간과 다른 시간에 거래가 발생할 경우 이를 이상거래로 간주할 수 있다.	새벽 0시부터 5시 사이의 시간에 거래가 발생했을 때, 이전에 해당 카드가 같은 시간대에 결제한 기록이 없을 경우 이상으로 탐지한다.
5	비정상적인 결제 패턴. 고객이 일반적으로 용하지 않는 상점에서의 거래 또는 이전에 이상거래가 발생한 상점에서의 거래가 발생할 경우 이를 이상거래로 간주할 수 있다.	해당 카드로 이전에 이용하지 않은 상점에서 결제할 경우 또는 이전에 이상거래가 발생한 기록이 있는 상점에서 결제할 경우 이상으로 탐지한다.

## 08. FDS요구사항

상품권 FDS		
번호	요인	해결
1	무작위 번호 입력	유효하지 않은 상품권 번호를 지속적으로 입력할 경우 이상거래로 탐지
2	제한된 결제수단 사용	조회한 상품권 번호의 상태가 이상거래 결제수단인 경우 이상거래로 탐지
3	해킹 및 명의도용	평소 결제패턴(평균 결제금액, 평균 결제횟수)에서 벗어난 결제시도가 감지될 경우 이상거래로 탐지
4	보이스피싱 및 타의에 의한 구매	상품권 번호 활성화 직후 짧은 시간내에 활성화된 위치와 멀리 떨어진 곳에서 결제시도가 감지될 경우 이상거래로 탐지
5	이상거래 주의 사용자에게 해당	동일한 사용자에게서 다수의 이상거래가 시도된 기록이 있을 경우 이상거래로 탐지

09. 물리설계

엔티티타입명	카드						
테이블명	CARDS						
번호	컬럼명	속성명	도메인	데이터 타입	길이	NULL 여부	KEY
1	CARD_NUM	카드번호	카드번호	VARCHAR2	16	NOT NULL	PK
2	CARD_HOLDER_NAME	카드소유자 이름	이름	VARCHAR2	30	NULL	
3	EXP_DATE	카드 유효기간	유효기간	VARCHAR2	7	NOT NULL	
4	CVC	CVC	CVC	VARCHAR2	3	NOT NULL	
5	CARD_COMPANY	카드사	카드사	VARCHAR2	20	NOT NULL	
엔티티타입명	결제						
테이블명	PAYMENTS						
번호	컬럼명	속성명	도메인	데이터 타입	길이	NULL 여부	KEY
1	PAYMENT_NUM	결제번호	번호	VARCHAR2	9	NOT NULL	PK

09. 물리설계

2	PRICE	가격	가격	NUMBER	10	NOT NULL	
3	PAYMENT_METHOD	결제수단	결제수단	VARCHAR2	20	NULL	
4	PAYMENT_TIME	결제시간	시간	TIMESTAMP		NOT NULL	
5	STORE_NAME	상점명	상점명	VARCHAR2	100	NOT NULL	
6	PAYMENT_STATUS	결제상태	상태	VARCHAR2	30	NOT NULL	
7	INSTALLMENTS	할부	할부	NUMBER	38	NULL	
8	IP_ADDR	IP주소	IP주소	VARCHAR2	25	NOT NULL	
9	CARD_NUM	카드번호	카드번호	VARCHAR2	50	NOT NULL	PK, FK
엔티티타입명	상품 권 결제내역						
테이블명	PAYMENT						
번호	컬럼명	속성명	도메인	데이터 타입	길이	NULL 여부	KEY
1	P_ID	결제번호	일련번호	VARCHAR	45	NOT NULL	PK

## 09. 물리설계

2	P_IP	접속 IP	IP 주소	VARCHAR	60	NOT NULL	
3	P_TIME	결제시간	시간	TIMESTAMP		NOT NULL	
4	P_STORE	가맹점명	내용	VARCHAR	100	NOT NULL	
5	P_ORDER	주문번호	일련번호	VARCHAR	45	NOT NULL	
6	P_NAME	사용자이름	이름	VARCHAR	80	NOT NULL	
7	P_PHONE	사용자전화번호	전화번호	VARCHAR	90	NOT NULL	
8	P_GOODS_NAME	상품명	내용	VARCHAR	100	NOT NULL	
9	P_GOODS_PRICE	상품금액	금액	NUMBER		NOT NULL	
10	P_METHOD	결제수단	내용	VARCHAR	100	NOT NULL	
11	P_GIFT_COMPANY	상품권유형	내용	VARCHAR	100	NOT NULL	
12	P_GIFT_CODE	상품권번호	일련번호	VARCHAR	45	NOT NULL	
13	P_GIFT_AMOUNT	상품권사용금액	금액	NUMBER		NULL	

09. 물리설계

14	P_TOTALPRICE	총 결제금액	금액	NUMBER		NULL	
15	P_STATUS	결제상태	내용	VARCHAR	100	NOT NULL	
16	P_FAILED_ID	이상거래번호	일련번호	VARCHAR	45	NULL	
17	P_FAILED_TITLE	이상거래사유	상세내용	VARCHAR	255	NULL	
18	P_FAILED_DESCRIPTION	이상거래내용	상세내용	VARCHAR	255	NULL	
엔티티타입명	번호 조회형 상품권						
테이블명	GIFT_CODE						
번호	컬럼명	속성명	도메인	데이터 타입	길이	NULL 여부	KEY
1	GC_CODE	상품권번호	일련번호	VARCHAR	45	NOT NULL	PK
2	GC_PRICE	상품권금액	금액	NUMBER		NOT NULL	
3	GC_EX_DATE	상품권만료일	날짜	DATE		NOT NULL	
4	GC_STATUS	상품권상태	내용	VARCHAR	100	NOT NULL	

09. 물리설계

5	GC_ACT_TIME	황성화시간	시간	TIMESTAMP		NULL	
6	GC_UP_TIME	업데이트시간	시간	TIMESTAMP		NULL	
7	GC_ACT_IP	활성화 IP	IP 주소	VARCHAR	60	NULL	
엔티티타입명	계정 연동형 상품						
테이블명	GIFT_ACCOUNT						
번호	컬럼명	속성명	도메인	데이터 타입	길이	NULL 여부	KEY
1	GA_ID	상품권계정아이디	아이디	VARCHAR	50	NOT NULL	PK
2	GA_PW	상품권계정비밀번호	내용	VARCHAR	100	NOT NULL	
3	GA_NAME	상품권계정이름	이름	VARCHAR	80	NOT NULL	
4	GA_PHONE	상품권계정전화번호	전화번호	VARCHAR	90	NOT NULL	
5	GA_AGE	상품권계정나이	나이	NUMBER		NOT NULL	
6	GA_BALANCE	상품권계정잔액	금액	NUMBER		NOT NULL	



09. 물리설계

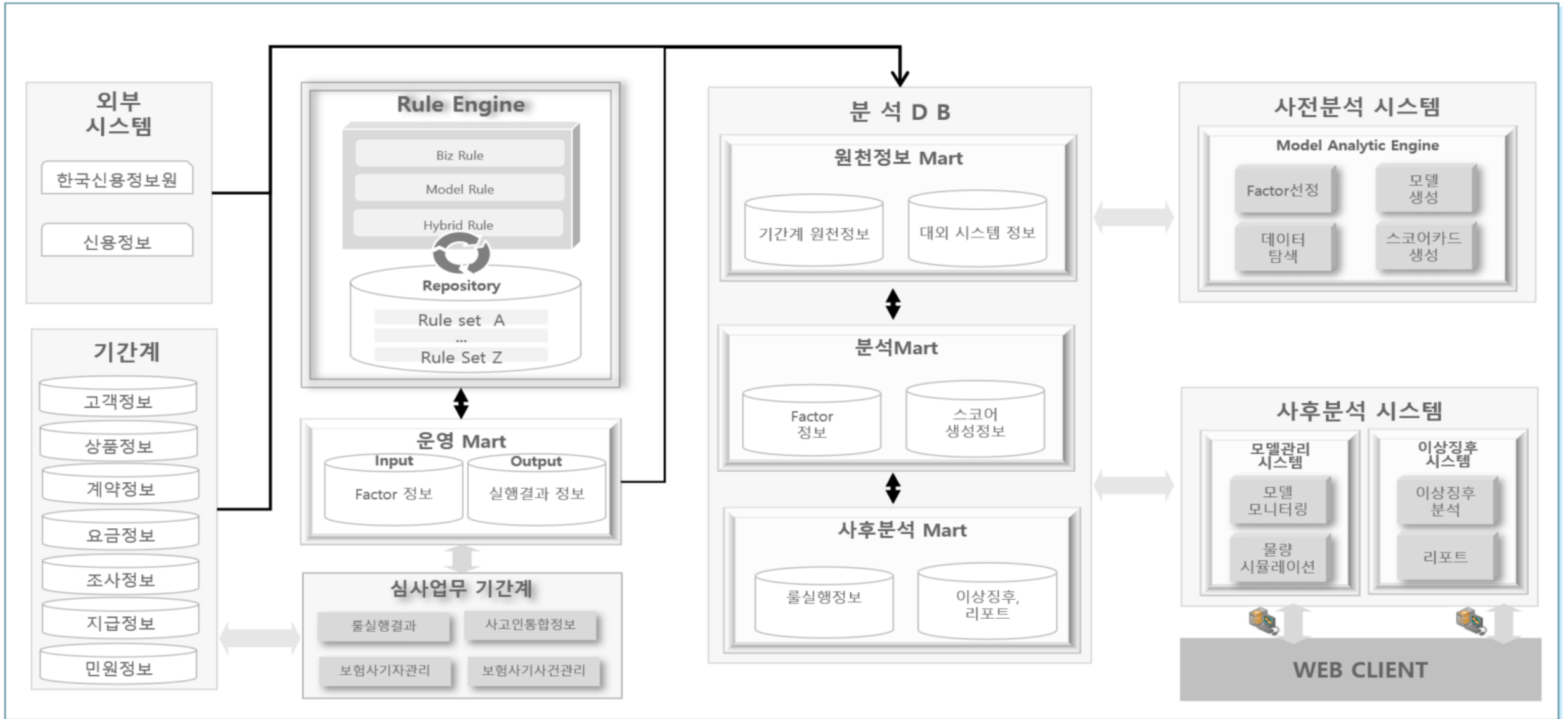
7	GA_JOIN_TIME	상품권계정가입날짜	시간	TIMESTAMP		NOT NULL	
8	GA_JOIN_IP	상품권계정가입 IP	IP 주소	VARCHAR	60	NOT NULL	
엔티티타입명	이상거래 항목						
테이블명	FRAUD						
번호	컬럼명	속성명	도메인	데이터 타입	길이	NULL 여부	KEY
1	F_ID	이상거래번호	일련번호	VARCHAR	45	NOT NULL	PK
2	F_TITLE	이상거래사유	상세내용	VARCHAR	255	NOT NULL	
3	F_DESCRIPTION	이상거래내용	상세내용	VARCHAR	255	NOT NULL	
엔티티타입명	이상거래 사용자 목록						
테이블명	FRUAD_USER						
번호	컬럼명	속성명	도메인	데이터 타입	길이	NULL 여부	KEY
1	USER_ID	사용자번호	일련번호	VARCHAR	45	NOT NULL	PK

09.

물리설계

2	USER_NAME	사용자이름	이름	VARCHAR	80	NOT NULL	
3	USER_PHONE	사용자전화번호	전화번호	VARCHAR	90	NOT NULL	
4	USER_RECENT_IP	사용자최근접속 IP	IP 주소	VARCHAR	60	NOT NULL	

# 10. 시스템 아키텍처



## 11. 기능명세서

어플리케이션	페이지명	파일명	요소	상세내용	개발	테스트
신용카드 결제	카드 정보 입력 페이지	cardForm.html	form	카드를 등록하기 위해 입력한 정보를 전송		
			Input-text	카드번호를 입력		
			Input-text	카드소유자명을 입력		
			select	카드사를 선택		
			Input-month	유효기간을 month타입으로 입력 후 문자타입으로 DB에 저장		
			Input-number	CVC를 입력		
			Input-submit	'입력완료' 버튼. 클릭 시 form 실행. 카드번호와 카드소유자명, 카드사, 유효기간, CVC를 DB에 입력		
	결제 정보 입력 페이지	paymentform.html	form	결제를 위해 입력한 정보를 전송		
			Input-hidden	이전 페이지에서 입력한 카드번호		
			Input-text	결제 상점의 이름을 입력		

## 11. 기능명세서

어플리케이션	페이지명	파일명	요소	상세내용	개발	테스트
신용카드 결제	결제 정보 입력 페이지	paymentform.html	Input-number	해당 결제의 총 결제 금액을 입력		
			Input-number	해당 결제의 할부 기간을 입력		
			Input-submit	'결제하기' 버튼. 클릭 시 form 실행. 결제 매장과 결제 금액, 할부 기간, 카드번호를 DB에 입력		
	관리자 콘솔 페이지	information.html	canvas	거래 유형별 건수를 표시하는 막대형 차트		
			canvas	이상거래 유형 비율을 표시하는 원형 차트		
			table	거래 유형별 건수를 표시하는 테이블		
			table	전체 거래 내역을 표시하는 테이블		
			button	'전체 거래' 버튼. 클릭 시 전체 거래 내역 테이블 에서 전체 거래 내역을 모두 표시		
			button	'비정상 거래' 버튼. 클릭 시 전체 거래 내역 테이블 에서 비정상 거래의 내역만을 표시		

## 11. 기능명세서

어플리케이션	페이지명	파일명	요소	상세내용	개발	테스트
상품권 결제	결제정보 입력 페이지	index.html	form	결제정보 입력 시 이상거래 탐지 시나리오를 거치고 DB에 저장		
			input-text	결제정보에 필요한 사용자 이름, 전화번호, 가맹점명, 주문번호, 상품명, 상품권 번호 입력		
			input-number	고객이 결제할 금액, 상품권 사용 금액 입력		
			input-radio	결제수단 입력		
			select	상품권 유형 입력		
			button	입력한 상품권 번호 조회, 입력한 금액 사용		
			button-submit	‘결제’버튼 클릭 시 form 태그 실행		
	결과 페이지	resultPage.html	table	DB에서 가져온 결제 리스트를 반복해서 출력		
			button	결제 데이터별 상세페이지로 이동		
			canvas	결제 데이터를 기반으로 날짜 별 이상거래 항목 횟수, 이상거래 검거 횟수 그래프로 출력		

## 11. 기능명세서

어플리케이션	페이지명	파일명	요소	상세내용	개발	테스트
상품권 결제	결제내용 페이지	detailPage.html	table	DB에서 가져온 결제 리스트를 반복해서 출력 선택한 결제 데이터의 상세내용 출력		
			button	결제 데이터별 상세페이지로 이동		

# 12. 스토리보드

페이지 제목	카드 정보 입력	ID	C-01
화면 경로	신용카드 결제		

카드 정보 입력

1

카드번호

2

카드소유자명


3

카드사

현대카드 ▼

4

유효기간

----년 --월 

5

CVC

6

입력완료

	상세 설명
1	- 등록할 신용카드의 카드번호를 입력
2	- 신용카드의 카드소유자명을 입력
3	- 신용카드의 카드사를 목록 중에서 선택
4	- 신용카드의 유효기간을 선택
5	- 신용카드의 CVC를 입력
6	- '입력완료' 버튼 - 입력된 정보를 DB에 등록 - 결제 정보 입력 화면으로 이동



# 12. 스토리보드

페이지 제목	결제 정보 입력	ID	C-02
화면 경로	신용카드 결제 > 입력완료		

	상세 설명
1	- 결제할 매장의 매장명을 입력
2	- 결제할 금액을 입력
3	- 할부 기간을 입력 - 미입력 시 일시불로 등록
4	- '결제하기' 버튼 - 입력된 정보를 DB에 등록 - 결제 결과창으로 이동

결제 정보 입력

1

결제 매장

2

결제 금액

3

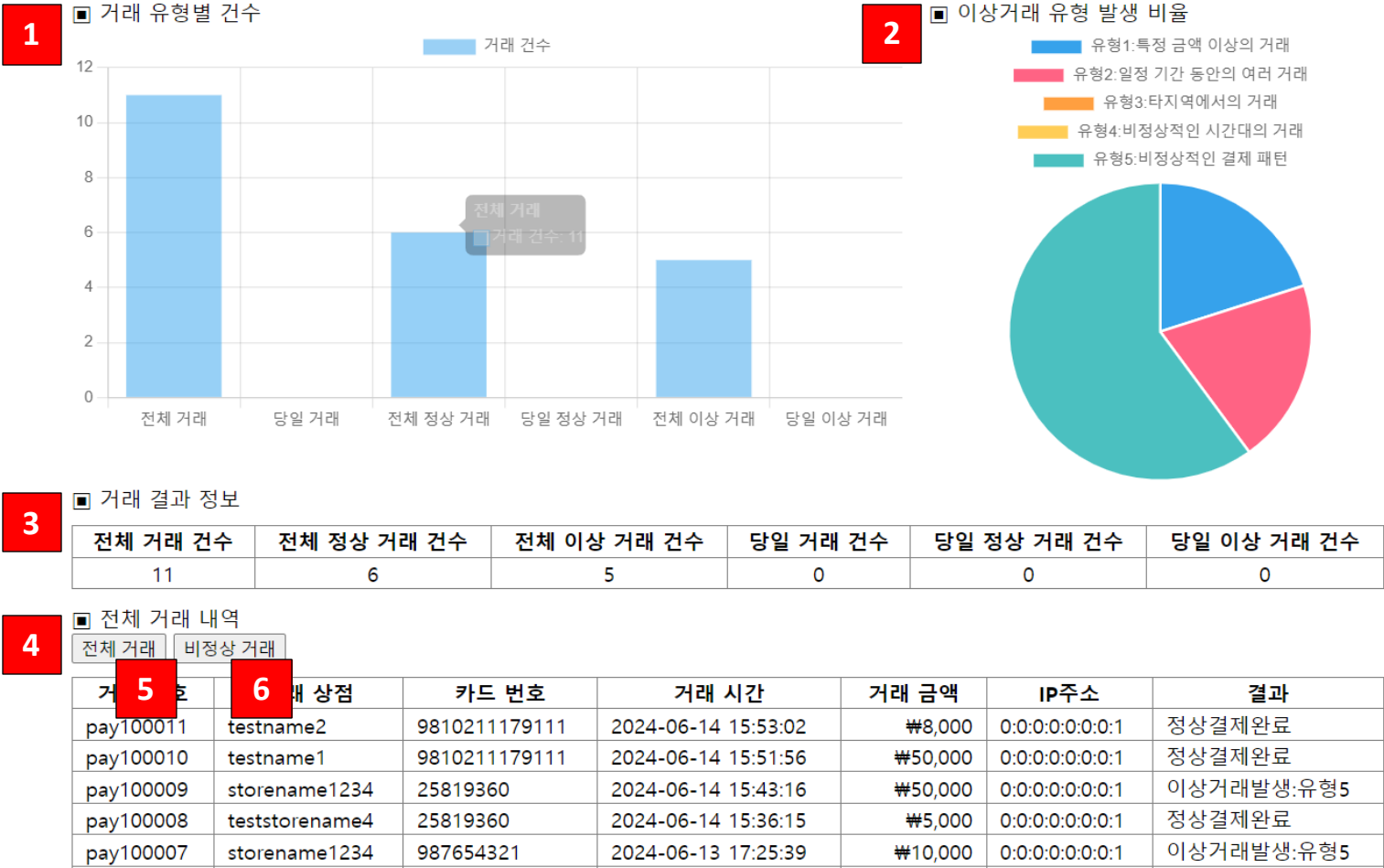
할부 기간

4

결제하기

# 12. 스토리보드

페이지 제목	관리자 콘솔 페이지	ID	I-01
화면 경로	거래 정보		



	상세 설명
1	- 거래 유형별 건수를 표시하는 막대형 차트
2	- 이상거래 유형 발생 비율을 표시하는 원형 차트
3	- 거래 유형별 건수를 표시하는 테이블
4	- 전체 거래 내역을 표시하는 테이블
5	- '전체 거래' 버튼 - '전체 거래 내역' 테이블에서 전체 거래 내역을 모두 표시
6	- '비정상 거래' 버튼 - '전체 거래 내역' 테이블에서 비정상 거래 내역만을 표시

# 12. 스토리보드

페이지 제목	상품권 결제	ID	G-01
화면 경로	상품권 결제		

	상세 설명
1	- 사용자의 결제정보 입력
2	- 결제항목 입력
3	- 이용약관 동의 확인
4	- 상품권유형 선택
5	- 상품권정보 입력
6	- '결제' 버튼 클릭 시 결과 페이지로 이동

1

결제정보

가맹점명

이름

전화번호

주요번호

2

결제항목

상품명

상품금액

상품권 사용

총 결제금액

- 0

0 원

3

이용약관 동의

☐ 전자금융거래 이용약관 개인정보의 수집 및 이용안내 (필수)

☐ 개인정보 제공 및 위탁안내 (필수)

4

결제수단

☐ 신용카드 / 체크카드

☐ 간편결제

☐ 유류카드

☒ 문화상품권 / 기프트카드

상품권 선택

스마트은상

상품권번호

4자리

4자리

4자리

6자리

조회

5

사용가능금액

사용

6

결제요청

# 12. 스토리보드

페이지 제목	결과	ID	G-02
화면 경로	상품권 결제 > 결과		

	상세 설명
1	- 저장된 결제 데이터의 일부를 출력
2	- '보기' 버튼 클릭 시 해당 데이터의 상세내용을 열람할 수 있는 페이지로 이동
3	- 이상거래로 간주된 결제데이터 항목별 차트 출력
4	- 날짜별 이상거래 탐지 횟수 차트 출력

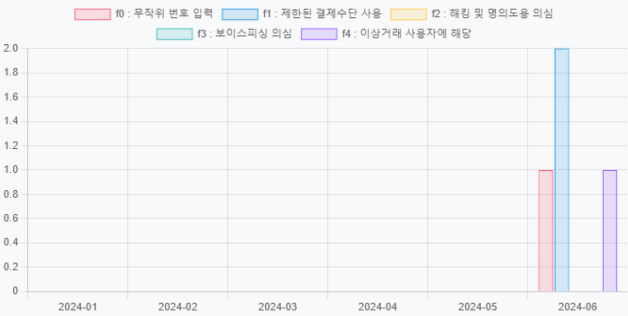
1

결제목록

번호	결제번호	날짜 / 시간	이름	전화번호	상태	상세내용
1	pid100003	2024-06-22 18:42:37	김민희	01047758616	2	<a href="#">보기</a>
2	pid100005	2024-06-24 13:09:32	김가영	01047418616	성공	<a href="#">보기</a>
3	pid100006	2024-06-24 13:13:51	김주용	01027578616	성공	<a href="#">보기</a>
4	pid100001	2024-06-22 18:18:05	김민희	01047758616	실패	<a href="#">보기</a>
5	pid100002	2024-06-22 18:19:21	김민희	01047758616	실패	<a href="#">보기</a>
6	pid100004	2024-06-22 18:51:16	김민희	01047758616	실패	<a href="#">보기</a>
7	pid100007	2024-06-25 14:30:19	김나라	01047859542	성공	<a href="#">보기</a>
8	pid100008	2024-06-25 14:35:38	이영애	01048596847	성공	<a href="#">보기</a>
9	pid100009	2024-06-25 14:49:59	김민직	01078459858	성공	<a href="#">보기</a>

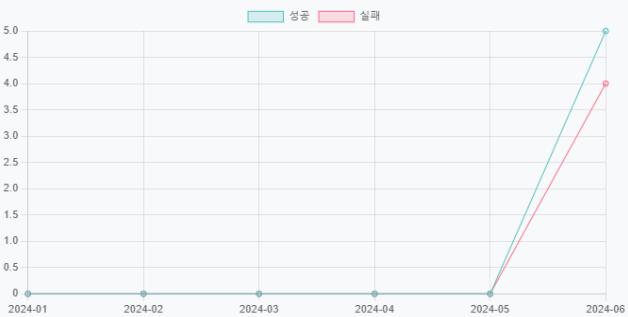
3

이상거래 항목별 탐지건수



4

날짜별 결제 성공여부



# 12. 스토리보드

페이지 제목	상세내용	ID	G-03
화면 경로	상품권 결제 > 결과 > 상세내용		

	상세 설명
1	- 저장된 결제 데이터의 일부를 출력
2	- '보기' 버튼 클릭 시 해당 데이터의 상세내용을 열람할 수 있는 페이지로 이동
3	- 선택된 데이터의 결제 데이터를 모두 출력
4	- 이상거래에 해당하는 경우 사유와 내용 출력

1

결제목록

번호	결제번호	날짜 / 시간	이름	전화번호	상태	상세내용
1	pid100003	2024-06-22 18:42:37	김민희	01047758616	2	<a href="#">보기</a>
2	pid100005	2024-06-24 13:09:32	김가영	01047418616	성공	<a href="#">보기</a>
3	pid100006	2024-06-24 13:13:51	김주홍	01027578616	성공	<a href="#">보기</a>
4	pid100001	2024-06-22 18:18:05	김민희	01047758616	실패	<a href="#">보기</a>
5	pid100002	2024-06-22 18:19:21	김민희	01047758616	실패	<a href="#">보기</a>
6	pid100004	2024-06-22 18:51:16	김민희	01047758616	실패	<a href="#">보기</a>
7	pid100007	2024-06-25 14:30:19	김나라	01047859542	성공	<a href="#">보기</a>
8	pid100008	2024-06-25 14:35:38	이영애	01048596847	성공	<a href="#">보기</a>

3

결제 상세내용

결제번호	pid100004
날짜 / 시간	2024-06-22 18:51:16
이름	김민희
전화번호	01047758616
접속IP	220.126.203.170
가맹점명	넷마블
주문번호	net_a_1050648
상품명	게릴머니 충전
상품금액	50000
결제수단	상품권
결제회사	스마트트러스트
상품권 정보	1234-1234-1234-0000004
상품권 사용금액	50000
총 결제금액	-
결제상태	실패

4

사유 : 이상거래 주의 사용자에 해당  
동일한 사용자에게서 다수의 이상거래 시도 감지

## 13. 소스코드

## 카드 정보 입력

```
1 <form action="payment" method="post">
view <table>
3   <caption>카드 정보 입력</caption>
4   <tr>
5     <td>카드번호 </td>
6     <td><input type="text" name="cardNum" /></td>
7   </tr>
8   <tr>
9     <td>카드소유자명 </td>
10    <td><input type="text" name="cardHolderName" /></td>
11  </tr>
12  <tr>
13    <td>카드사 </td>
14    <td>
15      <select name="cardCompany" id="cardCompany">
16        <option value="현대카드">현대카드</option>
17        <option value="삼성카드">삼성카드</option>
18        <option value="KB국민카드">KB국민카드</option>
19        <option value="비씨카드">비씨카드</option>
20        <option value="신한카드">신한카드</option>
21        <option value="NH농협카드">NH농협카드</option>
22        <option value="하나카드">하나카드</option>
23        <option value="씨티카드">씨티카드</option>
24        <option value="롯데카드">롯데카드</option>
25        <option value="그외">그외</option>
26      </select>
27    </td>
28  </tr>
29  <tr>
30    <td>유효기간 </td>
31    <td><input type="month" name="expDate" /></td>
32  </tr>
33  <tr>
34    <td>CVC </td>
35    <td><input type="number" name="cvc" /></td>
36  </tr>
37  <tr align="center">
38    <td colspan="2"><input type="submit" value="입력완료" /></td>
39  </tr>
40 </table>
controller @GetMapping("payment")
2 public String payment() {
3   return "thymeleaf/payment/cardForm";
4 }
5
6 @PostMapping("payment")
7 public String payment(CardCommand cardCommand, Model model) {
8   cardInsertService.execute(cardCommand, model);
9   return "thymeleaf/payment/paymentForm";
10 }
```

```
1 @Service
service public class CardInsertService {
2
3   @Autowired
4   CardRepository cardRepository;
5
6   public void execute(CardCommand cardCommand, Model
7     CardDTO dto = new CardDTO();
8     BeanUtils.copyProperties(cardCommand, dto);
9     cardRepository.cardInsert(dto);
10    model.addAttribute("cardNum", dto.getCardNum());
11  }
12
13  @Mapper(namespace="cardRepositorySql")
mapper <sql id="cardBaseColumns">
2    card_num, card_holder_name, exp_date, cvc, card_company
3  </sql>
4
5  <update id="cardInsert" parameterType="card">
6    merge into cards c
7    using dual
8    on (c.card_num = #{cardNum})
9    when matched then
10      update set card_holder_name = #{cardHolderName}, exp_date = #{expDate}
11              , cvc = #{cvc}, card_company = #{cardCompany}
12    when not matched then
13      insert (<include refid="cardBaseColumns" />)
14      values (#{cardNum}, #{cardHolderName}, #{expDate}, #{cvc}, #{cardCompany})
15  </update>
16 </mapper>
```

```
1 @Repository
repository public class CardRepository {
2
3   @Autowired
4   SqlSession sqlSession;
5
6   String namespace = "cardRepositorySql";
7   String statement;
8
9   public int cardInsert(CardDTO dto) {
10     statement = namespace + ".cardInsert";
11     return sqlSession.update(statement, dto);
12   }
13 }
```

# 13. 소스코드

## 신용카드 결제 정보 입력

```
1 <form action="paymentOk" method="post">
2 view <input type="hidden" name="cardNum" th:value="${cardNum}" />
3 <table>
4 <caption>결제 정보 입력</caption>
5 <tr>
6 <td>결제 매장</td>
7 <td><input type="text" name="storeName" /></td>
8 </tr>
9 <tr>
10 <td>결제 금액</td>
11 <td><input type="number" name="price" /></td>
12 </tr>
13 <tr>
14 <td>할부 기간</td>
15 <td><input type="number" name="installments" value="0" /></td>
16 </tr>
17 <tr align="center">
18 <td colspan="2">
19 <input type="submit" value="결제하기" />
20 </td>
21 </tr>
22 </table>
23 </form>
```

```
1 @Repository("springBootFinalProject.mapper.PaymentMapper")
2 public interface PaymentMapper {
3     public int paymentInsert(PaymentDTO dto);
4     public List<PaymentDTO> paymentSelectList();
5     public List<PaymentDTO> abnormalPaymentSelectList();
6     public List<Integer> paymentSelectPrice(String cardNum);
7     public Timestamp paymentSelectMaxTime(String cardNum);
8     public Timestamp paymentSelectSecondTime(String cardNum);
9     public int paymentStatusUpdate(String paymentStatus);
10    public String paymentSelectIp(String cardNum);
11    public List<Timestamp> paymentSelectTime(String cardNum);
12    public List<String> paymentSelectAbnormalStore();
13    public List<String> paymentSelectUsedStore(String cardNum);
14 }
```

```
1 @PostMapping("paymentOk")
2 controller String paymentOk(PaymentCommand paymentCommand, Model m
3     paymentInsertService.execute(paymentCommand, model, request)
4     return "thymeleaf/payment/buyfinished";
5 }
```

```
1 @Service
2 public class PaymentInsertService {
3     @Autowired PaymentMapper;
4     public void execute(PaymentCommand paymentCommand, Model model, HttpServletRequest request) {
5         PaymentDTO dto = new PaymentDTO();
6         BeanUtils.copyProperties(paymentCommand, dto);
7
8         String ipAddr = request.getRemoteAddr();
9         dto.setIpAddr(ipAddr);
10        String ipAddrBefore = paymentMapper.paymentSelectIp(dto.getCardNum()); // 최근 ip주소
11
12        Timestamp timestamp = new Timestamp(System.currentTimeMillis()); // 현재 시간
13        Timestamp firstTime = paymentMapper.paymentSelectMaxTime(dto.getCardNum());
14        Timestamp secondTime = paymentMapper.paymentSelectSecondTime(dto.getCardNum());
15
16        // 타임스탬프에서 시간 추출
17        Calendar calendar = Calendar.getInstance();
18        calendar.setTimeInMillis(timestamp.getTime());
19        int hour = calendar.get(Calendar.HOUR_OF_DAY);
20
21        if (dto.getPrice() > 500000) {
22            List<Integer> list = paymentMapper.paymentSelectPrice(dto.getCardNum());
23            int check = 0;
24            for (int price : list) {
25                if (price > 500000) {
26                    check++;
27                }
28            }
29            if (check == 0) {
30                model.addAttribute("paymentStatus", "이상거래발생:유형1");
31            } else {
32                model.addAttribute("paymentStatus", "정상결제완료");
33            }
34        } else if (firstTime != null && Math.abs(timestamp.getTime() - firstTime.getTime()) <= 300000) {
35            if (secondTime != null && Math.abs(firstTime.getTime() - secondTime.getTime()) <= 300000) {
36                model.addAttribute("paymentStatus", "이상거래발생:유형2");
37            } else {
38                model.addAttribute("paymentStatus", "정상결제완료");
39            }
40        } else if (ipAddrBefore != null && !ipAddr.equals(ipAddrBefore)) {
41            model.addAttribute("paymentStatus", "이상거래발생:유형3");
42        } else if (hour <= 5) { // 0시부터 5시까지
43            List<Timestamp> list = paymentMapper.paymentSelectTime(dto.getCardNum());
44            int check = 0;
45            for (Timestamp beforeTime : list) {
46                if (beforeTime != null) {
47                    calendar = Calendar.getInstance();
48                    calendar.setTimeInMillis(beforeTime.getTime());
49                    int beforeHour = calendar.get(Calendar.HOUR_OF_DAY);
50                    if (beforeHour <= 5) {
51                        check++;
52                    }
53                }
54            }
55            if (check == 0) {
56                model.addAttribute("paymentStatus", "이상거래발생:유형4");
57            } else {
58                model.addAttribute("paymentStatus", "정상결제완료");
59            }
60        } else {
61            int check = 0;
62            List<String> usedStoreNames = paymentMapper.paymentSelectUsedStore(dto.getCardNum());
63            List<String> abnormalStoreNames = paymentMapper.paymentSelectAbnormalStore();
64            for (String abnormalStoreName : abnormalStoreNames) {
65                if (dto.getStoreName().equals(abnormalStoreName)) {
66                    check++;
67                }
68            }
69            if (check != 0) {
70                model.addAttribute("paymentStatus", "이상거래발생:유형5");
71            } else {
72                model.addAttribute("paymentStatus", "정상결제완료");
73            }
74        }
75        paymentMapper.paymentInsert(dto);
76    }
77 }
78 }
79 }
80 }
81 }
```

```
1 @Mapper
2 public interface PaymentMapper {
3     <sql id="paymentBaseColumns">
4         num, price, payment_method, payment_time, store_name, payment_status, card_num, INSTALLMENTS, ip_addr
5     </sql>
6     <sql id="autoPaymentNum">
7         select concat('pay', nvl(max(substr(payment_num, 4)), 100000) + 1) from payments
8     </sql>
9     <insert id="paymentInsert" parameterType="payment">
10        insert into payments(<include refid="paymentBaseColumns" />)
11        values(<include refid="autoPaymentNum" />, #{price}, '신용카드', systimestamp, #{storeName},
12            '정상결제완료', #{cardNum}, #{installments}, #{ipAddr})
13    </insert>
14    <select id="paymentSelectList" resultType="payment">
15        select <include refid="paymentBaseColumns" />
16        from payments
17        order by payment_num desc
18    </select>
19    <select id="abnormalPaymentSelectList" resultType="payment">
20        select <include refid="paymentBaseColumns" />
21        from payments
22        where payment_status != '정상결제완료'
23        order by payment_num desc
24    </select>
25    <select id="paymentSelectPrice" parameterType="string" resultType="integer">
26        select price
27        from payments
28        where card_num = #{cardNum}
29    </select>
30    <select id="paymentSelectMaxTime" parameterType="string" resultType="java.sql.Timestamp">
31        select max(payment_time)
32        from payments
33        where card_num = #{cardNum}
34    </select>
35    <select id="paymentSelectSecondTime" parameterType="string" resultType="java.sql.Timestamp">
36        SELECT MAX(payment_time)
37        FROM payments
38        WHERE card_num = #{cardNum} and payment_time NOT IN (select max(payment_time) from payments where card_num = #{cardNum})
39    </select>
40    <update id="paymentStatusUpdate" parameterType="string">
41        update payments
42        set payment_status = #{paymentStatus}
43        where payment_num = (select payment_num
44            from payments
45            where substr(payment_num, 4) = (select max(substr(payment_num, 4)) from payments))
46    </update>
47    <select id="paymentSelectIp" parameterType="string" resultType="string">
48        select ip_addr
49        from payments
50        where card_num = #{cardNum}
51        order by payment_num desc
52        FETCH FIRST 1 ROW ONLY
53    </select>
54    <select id="paymentSelectTime" parameterType="string" resultType="java.sql.Timestamp">
55        select payment_time
56        from payments
57        where card_num = #{cardNum}
58    </select>
59    <select id="paymentSelectAbnormalStore" resultType="string">
60        select store_name
61        from payments
62        where payment_status != '정상결제완료'
63    </select>
64    <select id="paymentSelectUsedStore" parameterType="string" resultType="string">
65        select store_name
66        from payments
67        where card_num = #{cardNum}
68    </select>
69 }
```



2 **view** `<tr>`  
3  
4

```
1 @GetMapping("paymentDetail")
2 public String paymentDetail(Model model, @Request
3     paymentListService.execute(model, item);
4     return "thymeleaf/payment/information";
5 }
```

**service**

```

41 // 이상거래 유형4 발생 건수
42 count = paymentMapper.paymentCountCategory4();
43 model.addAttribute("countCategory4", count);
44 // 이상거래 유형5 발생 건수
45 count = paymentMapper.paymentCountCategory5();
46 model.addAttribute("countCategory5", count);
47 }

```

## repository

```

45     <select id="paymentCountCategory4" resultType="integer">
46         select count(*)
47         from payments
48         where payment_status = '이심거래발생:유형4'
49     </select>
50     <select id="paymentCountCategory5" resultType="integer">
51         select count(*)
52         from payments
53         where payment_status = '이심거래발생:유형5'
54     </select>
55 </mapper>

```

©Saebyeol Yu. Saebyeol's PowerPoint



# 13. 소스코드

## 결제 정보 입력

### view

```
239<body class="bg-light">
240
241<div class="container">
242<div class="py-5 text-center">
243
244</div>
245
246<!-- form -->
247<form action="/check/check" method="post" th:object="${paymentCommand}" id="frm">
248
249<!-- hidden 목록 -->
250<input type="hidden" name="pId" th:value="${paymentCommand.pId}"/>
251<input type="hidden" id="pIp" name="pIp" th:value="${paymentCommand.pIp}"/>
252<input type="hidden" id="pGiftCode" name="pGiftCode" th:value="${paymentCommand.pGiftCode}"/>
253<input type="hidden" id="pGiftAmount" name="pGiftAmount" th:value="${paymentCommand.pGiftAmount}"/>
254<input type="hidden" id="pTotalPrice" name="pTotalPrice" th:value="${paymentCommand.pTotalPrice}"/>
255<input type="hidden" id="pFid" name="pFid" th:value="${paymentCommand.pFid}"/>
256
257<div class="row g-5">
258<div class="col-md-5 col-lg-4 order-md-last">
259
260<!-- 금액 : 결제항목에 대한 정보 작성 -->
261<div class="d-flex justify-content-between align-items-center mb-3">
262<span class="text-primary">결제항목</span>
263<span class="badge bg-primary rounded-pill">1</span>
264</div>
265<ul class="list-group mb-3">
266
267<!-- 결제 입력 -->
268<li class="list-group-item d-flex justify-content-between lh-sm">
269<div>
270<div class="h6 goods">상품명</div>
271<div style="font-weight: bold;">상품금액</div>
272</div>
273<div>
274<input type="text" class="input-goods" id="pGoodsName" name="pGoodsName" th:value="${pIp}" />
275<input type="text" class="input-goods" id="pGoodsPrice" name="pGoodsPrice" th:value="${pIp}" />
276</div>
277</li>
278
279<!-- 상품권 -->
280<li class="list-group-item d-flex justify-content-between bg-light">
281<div class="text-success">
282<div class="h6 goods">상품권 사용</div>
283<div style="font-weight: bold;">상품권 사용</div>
284</div>
285<div class="text-success" id="usePrice"> 0</div>
286</li>
287
288</div>
289</div>
290</div>
291</div>
292</div>
293</div>
294</div>
295</div>
296</div>
297</div>
298</div>
299</div>
300</div>
301</div>
302</div>
303</div>
304</div>
305</div>
306</div>
307</div>
308</div>
309</div>
310</div>
311</div>
312</div>
313</div>
314</div>
315</div>
316</div>
317</div>
318</div>
319</div>
320</div>
321</div>
322</div>
323</div>
324</div>
325</div>
326</div>
327</div>
328</div>
329</div>
330</div>
331</div>
332</div>
333</div>
334</div>
335</div>
336</div>
337</div>
338</div>
339</div>
340</div>
341</div>
342</div>
343</div>
344</div>
345</div>
346</div>
347</div>
348</div>
349</div>
350</div>
351</div>
352</div>
353</div>
354</div>
355</div>
356</div>
357</div>
358</div>
359</div>
360</div>
361</div>
362</div>
363</div>
364</div>
365</div>
366</div>
367</div>
368</div>
369</div>
370</div>
371</div>
372</div>
373</div>
374</div>
375</div>
376</div>
377</div>
378</div>
379</div>
380</div>
381</div>
382</div>
383</div>
384</div>
385</div>
386</div>
387</div>
388</div>
389</div>
390</div>
391</div>
392</div>
393</div>
394</div>
395</div>
396</div>
397</div>
398</div>
399</div>
400</div>
401</div>
402</div>
403</div>
404</div>
405</div>
406</div>
407</div>
408</div>
409</div>
410</div>
411</div>
412</div>
413</div>
414</div>
415</div>
416</div>
417</div>
418</div>
419</div>
420</div>
421</div>
422</div>
423</div>
424</div>
425</div>
426</div>
427</div>
428</div>
429</div>
430</div>
431</div>
432</div>
433</div>
434</div>
435</div>
436</div>
437</div>
438</div>
439</div>
440</div>
441</div>
442</div>
443</div>
444</div>
445</div>
446</div>
447</div>
448</div>
449</div>
450</div>
451</div>
452</div>
453</div>
454</div>
455</div>
456</div>
457</div>
458</div>
459</div>
460</div>
461</div>
462</div>
463</div>
464</div>
465</div>
466</div>
467</div>
468</div>
469</div>
470</div>
471</div>
472</div>
473</div>
474</div>
475</div>
476</div>
477</div>
478</div>
479</div>
480</div>
481</div>
482</div>
483</div>
484</div>
485</div>
486</div>
487</div>
488</div>
489</div>
490</div>
491</div>
492</div>
493</div>
494</div>
495</div>
496</div>
497</div>
498</div>
499</div>
500</div>
501</div>
502</div>
503</div>
504</div>
505</div>
506</div>
507</div>
508</div>
509</div>
510</div>
511</div>
512</div>
513</div>
514</div>
515</div>
516</div>
517</div>
518</div>
519</div>
520</div>
521</div>
522</div>
523</div>
524</div>
525</div>
526</div>
527</div>
528</div>
529</div>
530</div>
531</div>
532</div>
533</div>
534</div>
535</div>
536</div>
537</div>
538</div>
539</div>
540</div>
541</div>
542</div>
543</div>
544</div>
545</div>
546</div>
547</div>
548</div>
549</div>
550</div>
551</div>
552</div>
553</div>
554</div>
555</div>
556</div>
557</div>
558</div>
559</div>
560</div>
561</div>
562</div>
563</div>
564</div>
565</div>
566</div>
567</div>
568</div>
569</div>
570</div>
571</div>
572</div>
573</div>
574</div>
575</div>
576</div>
577</div>
578</div>
579</div>
580</div>
581</div>
582</div>
583</div>
584</div>
585</div>
586</div>
587</div>
588</div>
589</div>
590</div>
591</div>
592</div>
593</div>
594</div>
595</div>
596</div>
597</div>
598</div>
599</div>
600</div>
601</div>
602</div>
603</div>
604</div>
605</div>
606</div>
607</div>
608</div>
609</div>
610</div>
611</div>
612</div>
613</div>
614</div>
615</div>
616</div>
617</div>
618</div>
619</div>
620</div>
621</div>
622</div>
623</div>
624</div>
625</div>
626</div>
627</div>
628</div>
629</div>
630</div>
631</div>
632</div>
633</div>
634</div>
635</div>
636</div>
637</div>
638</div>
639</div>
640</div>
641</div>
642</div>
643</div>
644</div>
645</div>
646</div>
647</div>
648</div>
649</div>
650</div>
651</div>
652</div>
653</div>
654</div>
655</div>
656</div>
657</div>
658</div>
659</div>
660</div>
661</div>
662</div>
663</div>
664</div>
665</div>
666</div>
667</div>
668</div>
669</div>
670</div>
671</div>
672</div>
673</div>
674</div>
675</div>
676</div>
677</div>
678</div>
679</div>
680</div>
681</div>
682</div>
683</div>
684</div>
685</div>
686</div>
687</div>
688</div>
689</div>
690</div>
691</div>
692</div>
693</div>
694</div>
695</div>
696</div>
697</div>
698</div>
699</div>
700</div>
701</div>
702</div>
703</div>
704</div>
705</div>
706</div>
707</div>
708</div>
709</div>
710</div>
711</div>
712</div>
713</div>
714</div>
715</div>
716</div>
717</div>
718</div>
719</div>
720</div>
721</div>
722</div>
723</div>
724</div>
725</div>
726</div>
727</div>
728</div>
729</div>
730</div>
731</div>
732</div>
733</div>
734</div>
735</div>
736</div>
737</div>
738</div>
739</div>
740</div>
741</div>
742</div>
743</div>
744</div>
745</div>
746</div>
747</div>
748</div>
749</div>
750</div>
751</div>
752</div>
753</div>
754</div>
755</div>
756</div>
757</div>
758</div>
759</div>
760</div>
761</div>
762</div>
763</div>
764</div>
765</div>
766</div>
767</div>
768</div>
769</div>
770</div>
771</div>
772</div>
773</div>
774</div>
775</div>
776</div>
777</div>
778</div>
779</div>
780</div>
781</div>
782</div>
783</div>
784</div>
785</div>
786</div>
787</div>
788</div>
789</div>
790</div>
791</div>
792</div>
793</div>
794</div>
795</div>
796</div>
797</div>
798</div>
799</div>
800</div>
801</div>
802</div>
803</div>
804</div>
805</div>
806</div>
807</div>
808</div>
809</div>
810</div>
811</div>
812</div>
813</div>
814</div>
815</div>
816</div>
817</div>
818</div>
819</div>
820</div>
821</div>
822</div>
823</div>
824</div>
825</div>
826</div>
827</div>
828</div>
829</div>
830</div>
831</div>
832</div>
833</div>
834</div>
835</div>
836</div>
837</div>
838</div>
839</div>
840</div>
841</div>
842</div>
843</div>
844</div>
845</div>
846</div>
847</div>
848</div>
849</div>
850</div>
851</div>
852</div>
853</div>
854</div>
855</div>
856</div>
857</div>
858</div>
859</div>
860</div>
861</div>
862</div>
863</div>
864</div>
865</div>
866</div>
867</div>
868</div>
869</div>
870</div>
871</div>
872</div>
873</div>
874</div>
875</div>
876</div>
877</div>
878</div>
879</div>
880</div>
881</div>
882</div>
883</div>
884</div>
885</div>
886</div>
887</div>
888</div>
889</div>
890</div>
891</div>
892</div>
893</div>
894</div>
895</div>
896</div>
897</div>
898</div>
899</div>
900</div>
901</div>
902</div>
903</div>
904</div>
905</div>
906</div>
907</div>
908</div>
909</div>
910</div>
911</div>
912</div>
913</div>
914</div>
915</div>
916</div>
917</div>
918</div>
919</div>
920</div>
921</div>
922</div>
923</div>
924</div>
925</div>
926</div>
927</div>
928</div>
929</div>
930</div>
931</div>
932</div>
933</div>
934</div>
935</div>
936</div>
937</div>
938</div>
939</div>
940</div>
941</div>
942</div>
943</div>
944</div>
945</div>
946</div>
947</div>
948</div>
949</div>
950</div>
951</div>
952</div>
953</div>
954</div>
955</div>
956</div>
957</div>
958</div>
959</div>
960</div>
961</div>
962</div>
963</div>
964</div>
965</div>
966</div>
967</div>
968</div>
969</div>
970</div>
971</div>
972</div>
973</div>
974</div>
975</div>
976</div>
977</div>
978</div>
979</div>
980</div>
981</div>
982</div>
983</div>
984</div>
985</div>
986</div>
987</div>
988</div>
989</div>
990</div>
991</div>
992</div>
993</div>
994</div>
995</div>
996</div>
997</div>
998</div>
999</div>
1000</div>
```

### controller

```
15 @Controller
16 @RequestMapping("/application")
17 @MapperScan(value = "finalFDS.mapper")
18 public class FinalFdsApplication {
19
20 public static void main(String[] args) {
21 SpringApplication.run(FinalFdsApplication.class, args);
22 }
23
24 @Autowired
25 PaymentAutoNumService paymentAutoNumService;
26
27 @GetMapping("/")
28 public String index(PaymentCommand paymentCommand, Model model){
29 paymentAutoNumService.execute(model);
30 return "thymeleaf/index";
31 }
32
33 }
```

### service

```
14 @Service
15 public class PaymentInsertService {
16
17 @Autowired
18 PaymentMapper paymentMapper;
19 @Autowired
20 FraudMapper fraudMapper;
21
22 public void execute(PaymentCommand paymentCommand, String status, String pfid) {
23
24 PaymentDTO paymentDTO = new PaymentDTO();
25
26 if(status == "fail") {
27 FraudDTO fraudDTO = fraudMapper.fraudSelectOne(pfid);
28 paymentDTO.setPFid(fraudDTO.getPFid());
29 paymentDTO.setPFTitle(fraudDTO.getFTitle());
30 paymentDTO.setPFDescription(fraudDTO.getFDDescription());
31
32 }else if(status == "success") {
33 paymentDTO.setPFid("");
34 paymentDTO.setPFTitle("");
35 paymentDTO.setPFDescription("");
36
37 }
38
39 paymentDTO.setPGiftAmount(paymentCommand.getPGiftAmount());
40 paymentDTO.setPGiftCode(paymentCommand.getPGiftCode());
41 paymentDTO.setPGiftCompany(paymentCommand.getPGiftCompany());
42 paymentDTO.setPGoodsName(paymentCommand.getPGoodsName());
43 paymentDTO.setPGoodsPrice(paymentCommand.getPGoodsPrice());
44 paymentDTO.setPID(paymentCommand.getPID());
45 paymentDTO.setPIp(paymentCommand.getPIp());
46 paymentDTO.setPMethod(paymentCommand.getPMethod());
47 paymentDTO.setPName(paymentCommand.getPName());
48 paymentDTO.setPOrder(paymentCommand.getPOrder());
49 paymentDTO.setPPhone(paymentCommand.getPPhone());
50
51 }
52
53 }
54
55 }
56
57 }
58
59 }
60
61 }
62
63 }
64
65 }
66
67 }
68
69 }
70
71 }
72
73 }
74
75 }
76
77 }
78
79 }
80
81 }
82
83 }
84
85 }
86
87 }
88
89 }
90
91 }
92
93 }
94
95 }
96
97 }
98
99 }
100
101 }
102
103 }
104
105 }
106
107 }
108
109 }
110
111 }
112
113 }
114
115 }
116
117 }
118
119 }
120
121 }
122
123 }
124
125 }
126
127 }
128
129 }
130
131 }
132
133 }
134
135 }
136
137 }
138
139 }
140
141 }
142
143 }
144
145 }
146
147 }
148
149 }
150
151 }
152
153 }
154
155 }
156
157 }
158
159 }
160
161 }
162
163 }
164
165 }
166
167 }
168
169 }
170
171 }
172
173 }
174
175 }
176
177 }
178
179 }
180
181 }
182
183 }
184
185 }
186
187 }
188
189 }
190
191 }
192
193 }
194
195 }
196
197 }
198
199 }
200
201 }
202
203 }
204
205 }
206
207 }
208
209 }
210
211 }
212
213 }
214
215 }
216
217 }
218
219 }
220
221 }
222
223 }
224
225 }
```

### repository

```
12 @Repository(value = "finalFDS.mapper.PaymentMapper")
13 public interface PaymentMapper {
14 public List<PaymentDTO> paymentSelectList();
15 public void paymentInsert(PaymentDTO dto);
16 public int amountAvg(F2DTO f2dto);
17 public int countAvg(F2DTO f2dto);
18 public int countDay(F2DTO f2dto);
19 public int countFail(F2DTO f2dto);
20 public PaymentDTO paymentSelectOne(String pId);
21
22 public int countMonthFid(MonthFidDTO monthFidDTO);
23 public int countMonthStatus(MonthStatusDTO monthStatusDTO);
24 }
25 }
```

### mapper

```
78 <!-- 평균 결제 금액 -->
79 <select id="amountAvg" parameterType="f2" resultType="integer">
80 SELECT COALESCE(AVG(P_GIFT_AMOUNT),0) AS avg_amount
81 FROM payment
82 WHERE P_NAME = #{pName} AND P_PHONE = #{pPhone}
83 </select>
84
85 <!-- 평균 결제 횟수 -->
86 <select id="countAvg" parameterType="f2" resultType="integer">
87 SELECT COALESCE(AVG(payment_count),0) AS avg_payment_count
88 FROM ( SELECT to_char(P_TIME, 'yyyy-MM-dd') AS payment_date, COUNT(*) AS payment_count
89 FROM payment
90 WHERE p_name = #{pName} AND p_phone = #{pPhone}
91 GROUP BY to_char(P_TIME, 'yyyy-MM-dd'))
92 )
93 </select>
94
95 <!-- 당일 결제 횟수 -->
96 <select id="countDay" parameterType="f2" resultType="integer">
97 SELECT COUNT(*) AS payment_count
98 FROM payment
99 WHERE p_name = #{pName} AND p_phone = #{pPhone} AND to_char(P_TIME, 'yyyy-MM-dd') = #{pToday}
100 </select>
101
102 <!-- 실패 횟수 -->
103 <select id="countFail" parameterType="f2" resultType="integer">
104 SELECT COUNT(*) AS failCount
105 FROM payment
106 WHERE p_name = #{pName} AND p_phone = #{pPhone} AND P_STATUS = 'fail'
107 </select>
108
109 <select id="paymentSelectOne" resultMap="paymentResultMap" parameterType="string">
110 select <include refid="paymentBaseColumns"/>
111 from payment
112 where p_id = #{pId}
113 </select>
114
115 <select id="countMonthFid" parameterType="monthFid" resultType="integer">
116 SELECT COUNT(*)
117 FROM payment
118 WHERE TO_CHAR(P_TIME, 'YYYY-MM') = #{month} AND p_failed_id = #{fid}
119 </select>
120
121 <select id="countMonthStatus" parameterType="monthStatus" resultType="integer">
122 SELECT COUNT(*)
123 FROM payment
124 WHERE TO_CHAR(P_TIME, 'YYYY-MM') = #{month} AND p_status = #{status}
125 </select>
126
127 </mapper>
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771

```

# 13. 소스코드

## 결과 출력

```
235 <!-- body -->
236 <body class="bg-light">
237
238     <div class="div-parent">
239
240         <!-- 이미지 -->
241         <div class="py-4">
242             
243         </div>
244
245         <!-- 결제 리스트 -->
246         <div id="div-paymentlist">
247             <h3 id="title-list">결제목록</h3>
248             <table width="600" align="center" border="1">
249                 <thead>
250                     <tr>
251                         <th>번호</th>
252                         <th>결제번호</th>
253                         <th>날짜 / 시간</th>
254                         <th>이름</th>
255                         <th>전화번호</th>
256                         <th>상태</th>
257                     </tr>
258                     <th>상세내용</th>
259                 </thead>
260                 </thead>
261                 <tbody>
262                     <tr th:each="dto, idx : ${dtos}">
263                         <td>[[${idx.count}]]</td>
264                         <td>[[${dto.pId}]]</td>
265                         <td>[[${#dates.format(dto.pTime, 'yyyy-MM-dd HH:mm:ss')}]]</td>
266                         <td>[[${dto.pName}]]</td>
267                         <td>[[${dto.pPhone}]]</td>
268                         <td><span style="color: red;" th:if="${dto.pStatus == 'fail'}">실패</span>
269                             <span style="color: green;" th:if="${dto.pStatus == 'success'}">성공</span></td>
270                         <td><button type="button" th:attr="onClick=|btnDetail('${dto.pId}')|" style="width:90px;">상세보기</td>
271                     </tr>
272                 </tbody>
273             </table>
274         </div>
275
276         <!-- 그래프 -->
277         <div id="div-graph">
278             <h3 id="title-list">이상거래 항목별 합지건수</h3>
279             <canvas id="canvas-bar"></canvas>
280
281             <h3 id="title-list" style="margin-top: 40px;">날짜별 결제 성공여부</h3>
282             <canvas id="canvas-line" style="margin-bottom: 100px;"></canvas>
283         </div>
284     </div>
285
286 </body>
287 </html>
288
289 @GetMapping("check")
290 public String check(Model model) {
291     paymentService.execute(model);
292     paymentListService.execute(model);
293     return "thymeleaf/resultPage";
294 }
295
296 @PostMapping("check")
297 public String check(PaymentCommand paymentCommand, Model model) throws IOException, Geo{
298     // f0 or f1
299     String pfid = paymentCommand.getPfid();
300
301     if(pfid.equals("f0")) {
302         paymentInsertService.execute(paymentCommand, "fail", "f0");
303         fraudUserUpdateService.execute(paymentCommand);
304         chartService.execute(model);
305         paymentListService.execute(model);
306         return "thymeleaf/resultPage";
307     }
308 }
```

## controller

```
16 @Service
17 public class ChartService {
18
19     @Autowired
20     PaymentMapper paymentMapper;
21
22     public void execute(Model model) {
23
24         // HashMap 객체 생성
25         Map<String, List<Integer>> chart = new HashMap<>();
26         chart.put("monthF0", monthFid("f0"));
27         chart.put("monthF1", monthFid("f1"));
28         chart.put("monthF2", monthFid("f2"));
29         chart.put("monthF3", monthFid("f3"));
30         chart.put("monthF4", monthFid("f4"));
31         chart.put("monthSuccess", monthStatus("success"));
32         chart.put("monthFail", monthStatus("fail"));
33         System.out.println("dssfsfsfSF");
34         System.out.println(chart);
35         model.addAttribute("chart", chart);
36     }
37
38     public List<Integer> monthFid(String fId){
39
40         List<Integer> monthFidList = new ArrayList<Integer>();
41         monthFidList.add(monthFid("2024-01", fId));
42         monthFidList.add(monthFid("2024-02", fId));
43         monthFidList.add(monthFid("2024-03", fId));
44         monthFidList.add(monthFid("2024-04", fId));
45         monthFidList.add(monthFid("2024-05", fId));
46         monthFidList.add(monthFid("2024-06", fId));
47         return monthFidList;
48     }
49
50     @Repository(value = "finalFDS.mapper.FraudUserMapper")
51     public interface FraudUserMapper {
52         public void fraudUserInsert(FraudUserDTO dto);
53         public int fraudUserCheck(F2DTO f2dto);
54         public void fraudUserUpdate(FraudUserDTO dto);
55         public String fraudUserAutoId();
56     }
57 }
```

## repository

```
21 <!-- 결제 데이터에서 3번 이상 결제 실패로 조회 될 경우 사용자 목록에 저장 -->
22 <insert id="fraudUserInsert" parameterType="fraudUser">
23     insert into fraud_user(<include refid="FraudUserColumns"/>)
24     values( #{userId},
25             #{userName},
26             #{userPhone},
27             #{userRecentIp})
28 </insert>
29
30 <!-- 주의 사용자 목록에 있는지 조회 -->
31 <select id="fraudUserCheck" parameterType="f2" resultType="integer">
32     SELECT COUNT(*)
33     FROM Fraud_user
34     WHERE USER_NAME = #{pName} and USER_PHONE = #{pPhone}
35 </select>
36
37 <!-- 사용자 목록에 이미 있는 경우 업데이트 -->
38 <update id="fraudUserUpdate" parameterType="fraudUser">
39     update fraud_user
40     SET USER_RECENT_IP = #{userRecentIp}
41     WHERE USER_NAME = #{userName} and USER_PHONE = #{userPhone}
42 </update>
43
44 <!-- 고유번호 자동 생성기 -->
45 <select id="fraudUserAutoId" resultType="string">
46     select concat('user_', nvl(max(substr(user_id,6)),100000) + 1)
47     from fraud_user
48 </select>
49 </mapper>
50
51 <resultMap type="fraud" id="fraudResultMap">
52     <id column="F_ID" jdbcType="VARCHAR" property="fId"/>
53     <result column="F_TITLE" jdbcType="VARCHAR" property="fTitle"/>
54     <result column="F_DESCRIPTION" jdbcType="VARCHAR" property="fDescription"/>
55 </resultMap>
56
57 <select id="fraudSelectOne" parameterType="string" resultMap="fraudResultMap">
58     select <include refid="FraudColumns"/>
59     from fraud
60     where F_ID = #{fId}
61 </select>
62 </mapper>
```

## mapper

```
35</body>  
36<body class="view">  
37<div class="view">
```

```

236<div class="bg-1ight">
237<div class="div-parent">
238
239
240<!-- 이미지 -->
241<div class="py-4">
242
243</div>
244
245<!-- 결제 리스트 -->
246<div id="div-paymentlist">
247<h3 id="title-list">결제목록</h3>
248<table width="600" align="center" border="1">
249<thead>
250<tr>
251<th>번호</th>
252<th>결제번호</th>
253<th>날짜 / 시간</th>
254<th>이름</th>
255<th>전화번호</th>
256<th>성명</th>
257<th></th>
258<th>상세내용</th>
259</tr>
260</thead>
261<tbody>
262<tr>
263<th:each="dto, idx : ${dto}">
264<td>[[${idx.count}]]</td>
265<td>[[${dto.pid}]]</td>
266<td>[[${dates.format(dto.pTime, 'yyyy-MM-dd HH:mm:ss')}]]</td>
267<td>[[${dto.pName}]]</td>
268<td>[[${dto.pPhone}]]</td>
269<td><span style="color: red;" th:if="${dto.pStatus == 'fail'}">실패</span>
270<span style="color: green;" th:if="${dto.pStatus == 'success'}">성공</span></td>
271<td><button type="button" th:attr="onClick=|btnDetail('${dto.pId}')|" style="width:
272</td>
273</tbody>
274</table>
275</div>
276
277<!-- 그래프 -->
278<div class="div-paymentInfo">
279<div id="div-resultInfo">
280<h3 id="title-detail">결제 상세내용</h3>
281<table width="800" align="center" border="1" class="table-paymentOne">
282<tr><td>결제번호</td>
283<td>[[${paymentDTO.pId}]]</td></tr>
284<tr><td>날짜 / 시간</td>
285<td>[[${dates.format(paymentDTO.pTime, 'yyyy-MM-dd HH:mm:ss')}]]</td></tr>
286<tr><td>이름</td>
287<td>[[${paymentDTO.pName}]]</td></tr>
288<tr><td>전화번호</td>
289<td>[[${paymentDTO.pPhone}]]</td></tr>
290<tr><td>카드사</td>
291<td>[[${paymentDTO.pIp}]]</td></tr>
292<tr><td>가맹점명</td>
293<td>[[${paymentDTO.pStore}]]</td></tr>
294<tr><td>주문번호</td>
295<td>[[${paymentDTO.pOrder}]]</td></tr>
296<tr><td>상품명</td>
297<td>[[${paymentDTO.pGoodsName}]]</td></tr>
298<tr><td>상품코드</td>
299<td>[[${paymentDTO.pGoodsPrice}]]</td></tr>
300<tr><td>결제수단</td>
301<td><span th:if="${paymentDTO.pMethod == 'giftcard'}">상품권</span></td></tr>
302<tr><td>결제회사</td>
303<td><span th:if="${paymentDTO.pGiftCompany == 'code'}">카드회사</span>
304<span th:if="${paymentDTO.pGiftCompany == 'account'}">결제카드회사</span></td></tr>
305<tr><td>결제금액</td>
306<td>[[${paymentDTO.pGiftCode}]]</td></tr>
307<tr><td>상품권</td>
308<td>[[${paymentDTO.pGiftAmount}]]</td></tr>
309<tr><td>결제상태</td>
310<td><span th:if="${paymentDTO.pStatus == 'fail'}">실패</span>
311<span th:if="${paymentDTO.pStatus == 'success'}">성공</span></td></tr>
312<tr><td>결제금액</td>
313<td><span style="color: red;" th:if="${paymentDTO.pStatus == 'fail'}">실패</span>

```

service

```

42 public String execute(PaymentCommand paymentCommand) throws IOException {
43     // f2: 결제 및 영의도증 작성
44     String pName = paymentCommand.getPName(); // 결제를 시도하는 사용자 이름
45     String pPhone = paymentCommand.getPPhone(); // 결제를 시도하는 사용자 전화번호
46     Integer pGiftAmount = paymentCommand.getPGiftAmount(); // 결제를 시도하는 상품권 금액
47     LocalDate today = LocalDate.now();
48     String ptoday = today.format(DateTimeFormatter.ofPattern("yyyy-MM-dd")); // 결제를 시도하는 날짜
49     String pIp = paymentCommand.getPtIp(); // 결제를 시도하는 ip
50     String ipCountry = inIpOutCountry(pIp);
51     String ipCity = inIpOutCity(pIp);
52
53     // 확인
54     System.out.println("pIp" + pIp);
55     System.out.println("ipCountry" + ipCountry);
56     System.out.println("ipCity" + ipCity);
57
58     F2DTO f2dto = new F2DTO();
59     f2dto.setPName(pName);
60     f2dto.setPPhone(pPhone);
61     f2dto.setPToday(ptoday);
62
63     // 평균 결제 금액 이상
64     Integer amountAvg = paymentMapper.amountAvg(f2dto);
65     System.out.println("결제 금액" + pGiftAmount);
66     System.out.println("평균 결제 금액" + amountAvg*5);
67     if(amountAvg != 0 && pGiftAmount >= (amountAvg*5)) {
68         return "f2";
69     }
70
71     // 평균 결제 횟수 이상
72     System.out.println("ip 조회 : " + ipCountry);
73     // 해외 ip 조회
74     if(!ipCountry.equals("South Korea")) {
75         return "f2";
76     }
77
78     // f3 : 기프트 카드 활성화 직후 짧은 시간내에 활성화된 위치와 멀리 떨어진 곳에서 사용 시도
79     String pGiftCompany = paymentCommand.getPGiftCompany();
80     String pGiftCode = paymentCommand.getPGiftCode();
81     Date currentTime = new Date();
82     SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
83     String pTime = sdf.format(currentTime);
84
85     if(pGiftCompany == "code") {
86         // 결제 시도하려는 상품권 코드의 정보 : 활성화 시간, ip
87         return accountFoundService.execute(gaId);
88         return accountFoundService.execute(gaId);
89     }
90
91     // 결제 상세내역 확인
92     @GetMapping("Detail/{pId}")
93     public String Detail(@PathVariable("pId") String pId, Model model) {
94         paymentListService.execute(model);
95         paymentSelectService.execute(pId, model);
96         return "thymeleaf/detailPage";
97     }
98 }

```

```
mapper {
    SELECT COALESCE(A
```

```

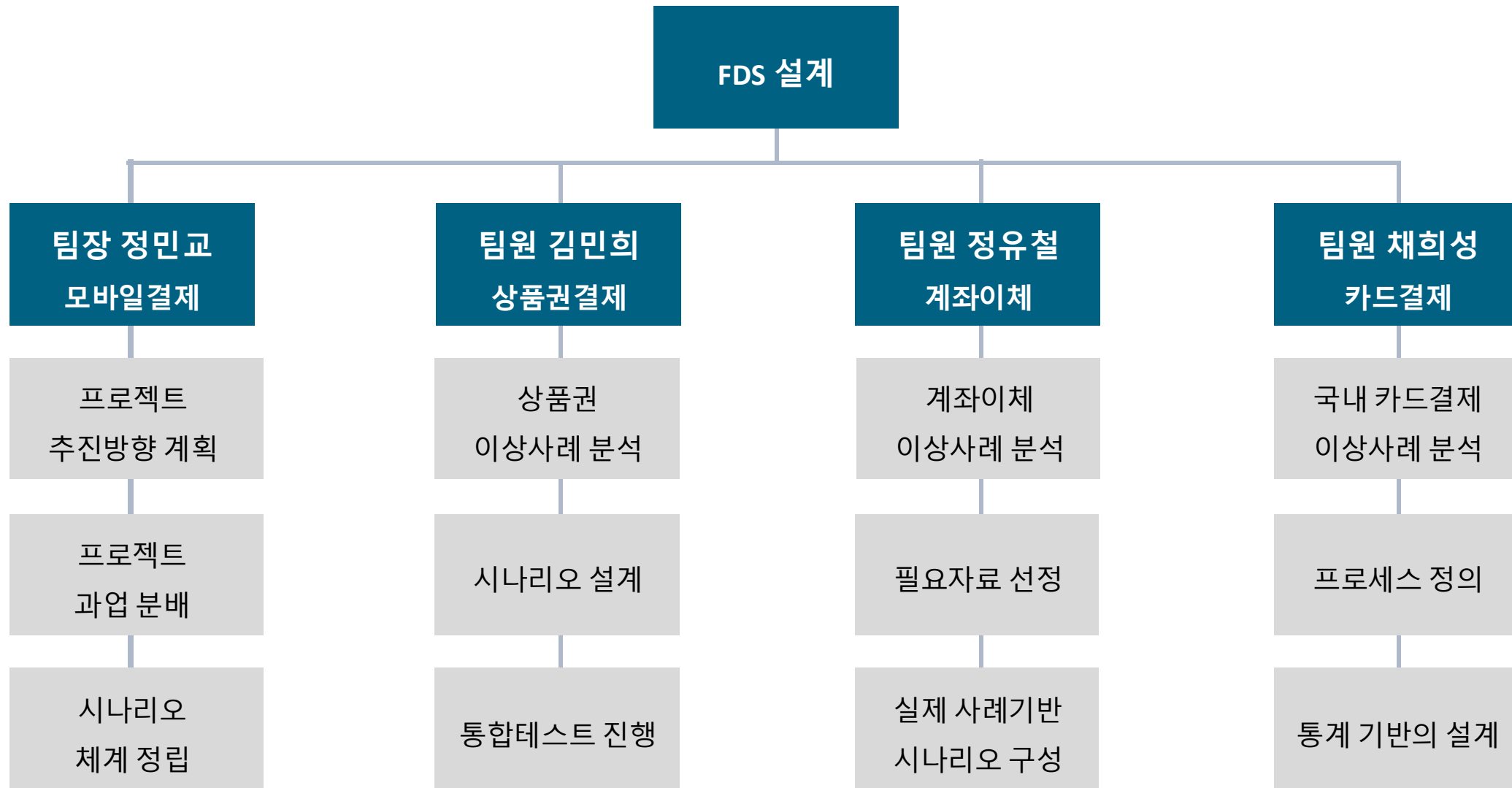
79<!-- 평균 결제 금액 -->
80<select id="countAvg" parameterType="f2" resultType="integer">
81    SELECT COALESCE(AVG(P_GIFT_AMOUNT),0) AS avg_amount
82    FROM payment
83    WHERE P_NAME = #{pName} AND P_PHONE = #{pPhone}
84</select>
85<!-- 평균 결제 횟수 -->
86<select id="countAvg" parameterType="f2" resultType="integer">
87    SELECT COALESCE(AVG(payment_count),0) AS avg_payment_count
88    FROM ( SELECT to_char(P_TIME, 'yyyy-MM-dd') AS payment_date, COUNT(*) AS payment_count
89           FROM payment
90           WHERE p_name = #{pName} AND p_phone = #{pPhone}
91           GROUP BY to_char(P_TIME, 'yyyy-MM-dd'))
92    )
93</select>
94<!-- 당일 결제 횟수 -->
95<select id="countDay" parameterType="f2" resultType="integer">
96    SELECT COUNT(*) AS payment_count
97    FROM payment
98    WHERE p_name = #{pName} AND p_phone = #{pPhone} AND to_char(P_TIME, 'yyyy-MM-dd') = #{pToday}
99</select>
100<!-- 실패 횟수 -->
101<select id="countFail" parameterType="f2" resultType="integer">
102    SELECT COUNT(*) AS failCount
103    FROM payment
104    WHERE p_name = #{pName} AND p_phone = #{pPhone} AND P_STATUS = 'fail'
105</select>
106<select id="paymentSelectOne" resultMap="paymentResultMap" parameterType="string">
107    select <include refid="paymentBaseColumns"/>
108    from payment
109    where p_id = #{pId}
110</select>
111
112@Repository(value = "finalFDS.mapper.PaymentMapper")
113public interface PaymentMapper {
114    public List<PaymentDTO> paymentSelectList();
115    public void paymentInsert(PaymentDTO dto);
116    public int amountAvg(F2DTO f2dto);
117    public int countAvg(F2DTO f2dto);
118    public int countDay(F2DTO f2dto);
119    public int countFail(F2DTO f2dto);
120    public PaymentDTO paymentSelectOne(String pId);
121
122    public int countMonthFid(MonthFidDTO monthFidDTO);
123    public int countMonthStatus(MonthStatusDTO monthStatusDTO)
124}
125

```

14.

# 일정(WBS)

타이틀	진행제목	진행자	5월			6월		
			2주차	3주차	4주차	1주차	2주차	3주차
개념설계	FDS 사례 탐색	팀 전체						
	FDS 시나리오 작성							
	시나리오 명사 도출							
	<u>엔티티</u> 타입 정의							
	관계 정의							
	속성 정의							
	식별자 정의							
	도메인 정의							
프로젝트 준비	프로젝트 기본 세팅	팀 전체						
	PPT 구성	채희성						
	발표자료	김민희/채희성						
	DB 구성	팀 전체						
신용카드	카드 정보 등록	채희성						
	결제 정보 등록							
	결제 결과 및 고객 알림							
	관리자 콘솔							
상품권	상품권 정보 등록	김민희						
	결제 정보 등록							
	결제 결과 및 자트 출력							
	결제 상세 내용 출력							
테스트	통합테스트	김민희/채희성						



16.

## 사용된 프로그램

