

Up Bank

은행 시스템 구축

한국소프트웨어 인재개발원 117기 / 김은진 김아람 백인준 이슬아 한채영 최슬기

Company's Growth

GLOBAL BUSINESS REVIEW

Mobile Phone & Tablet

Other Electronic Devices


Computers

Introduction

: 프로젝트 소개

UpBank는 JAVA와 Oracle DataBase를 기반으로 가볍고 직관적인 인터페이스를 통해 누구에게나 쉽게 다가갈 수 있는 은행을 목표로 한 Spring기반 웹사이트입니다.
Spring MVC패턴을 사용하여 유지보수가 쉽고,
Vue 등 신기술을 활용하여 차후 업데이트시에도 기술적 확장이 유연하도록 설계하였습니다.

Content

- 
- 01. 프로젝트 소개
 - 02. 프로젝트 일정
 - 03. 프로젝트 설계
 - 04. 프로젝트 시연
 - 05. 프로젝트 후기

About Team 팀원소개



김은진 팀장

프로젝트 총괄
형상관리(Git/Github)
계좌개설
환율 계산기 및 관리
환전관리
환율 API
JPA + Vue



김아람 부팀장

대출관리
대출신청 및 조회
대출납부
대출계산기
대출상환
관리자 - 대출 이자관리
JPA + Vue



최슬기 부팀장

예금관리
적금관리
계좌이체 및 이자계산기
만기시 이자지급
스케줄러
관리자 - 예금상품
JPA + Vue

About Team 팀원소개



한채영 팀원

Spring Security
로그인/회원가입
사용자계좌조회
계좌정보관리
관리자 통계
DB설계 및 ERD관리
JPA + Vue



이슬아 팀원

펀드리스트
펀드거래
펀드거래내역
고객센터
네이버파이낸스 api
펀드시세 및 거래량 차트
JPA + Vue



백인준 팀원

계좌이체
자동이체
거래내역 조회
AWS RDS 환경구축
Spring Boot 환경구축
스케줄러
JPA + Vue





EMP_PWD

admin1234



기존 암호화 방식은 노출 시 보안에 취약하여
BCrypt HASH를 통한 암호화 적용

EMP_PWD

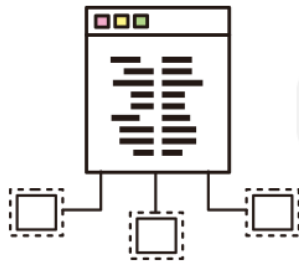
\$2a\$10\$YqYoK2wDy7I1wyM1NfooxuZlauCiQ
dizgLhLoxoU.2g3Lmb2U5r6S



Amazon RDS 인프라 및 데이터베이스 관리와 백업, 운영을
지원하는 클라우드 Amazon Web Service

JPA

- ✓ JPA는 애플리케이션과 JDBC 사이에 동작하는 자바의 ORM 표준 기술



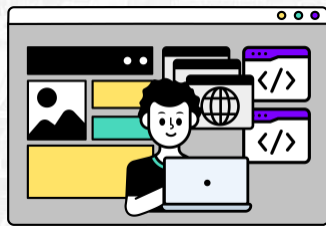
JPA

Java Persistence API

Vue

- ✓ Vue.js는 웹페이지 화면을 개발하기 위한 FrontEnd FrameWork로서 특징으로는 UI 화면 개발 방법 중 하나인 MVVM패턴의 뷰 모델에 해당하는 화면단의 라이브러리 Model-View-View Model로 구조화하여 개발하는 방식





유지보수의 편의성

MVVM패턴(Model-View-View Model)으로 구조화



유지보수의 편의성 증가

렌더링 속도

React와 Angular보다 빠른 렌더링 속도

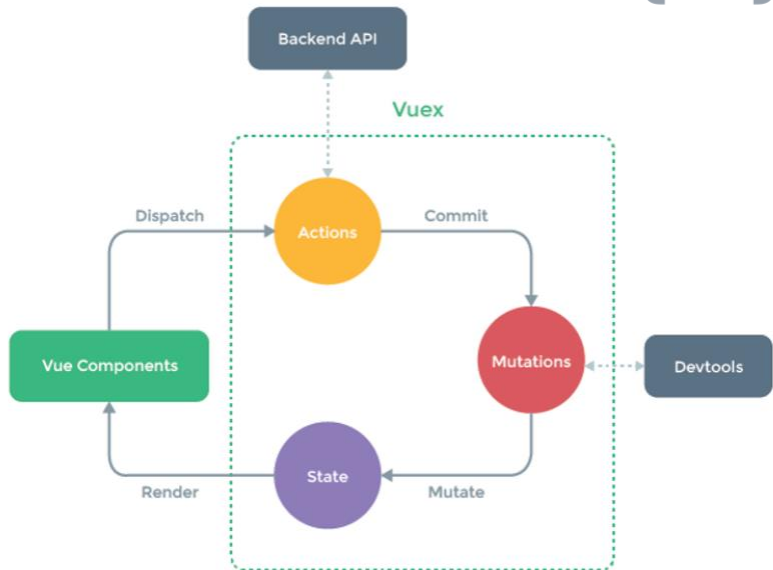
SPA (Single-Page Application)

서버로부터 완전한 새로운 페이지를 불러오지 않고
현재의 페이지를 동적으로 다시 작성하는,
사용자와 소통하는 웹 애플리케이션

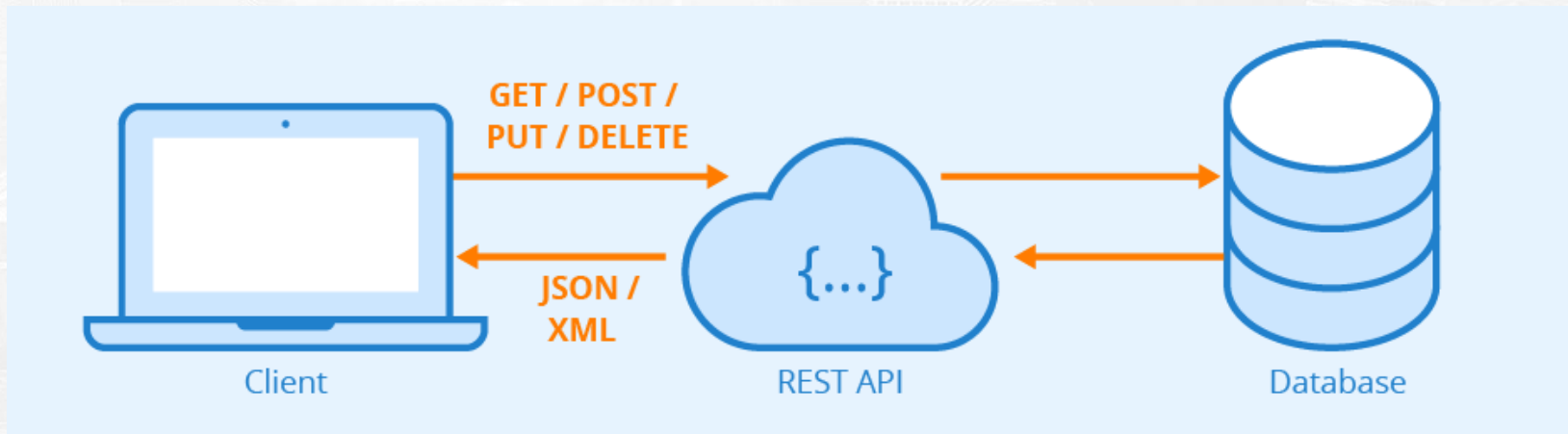


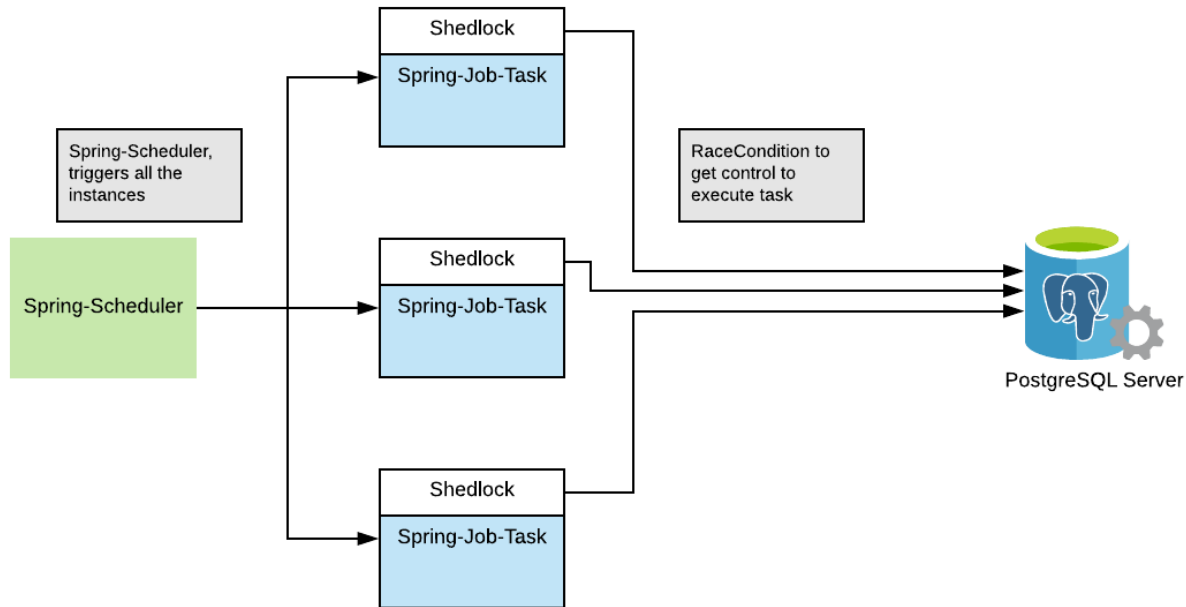
Vuex(store)

- State : Vue 컴포넌트에서 Data로 볼 수 있다.
원본 Data의 역할을 하며, View와 직접적으로 연결되어 있는 Model이다. 이 state는 직접적인 변경이 불가능하다.
- Mutations : State를 변경하는 유일한 방법이고, 함수로 구현되며, 첫 번째 인자는 State를 받고 두 번째 인자는 Payload를 받고. Commit으로 함수를 호출한다.
- Actions : Mutations를 실행시키는 역할을 담당하고, Mutations와 달리 비동기 작업이 가능하다. 첫 번째 인자는 Context를 받고 두 번째 인자는 Payload를 받는다.
- Getters : 서로 다른 Vue 컴포넌트에서 State의 값을 변경하지 않고 State를 호출만 하도록 한다.



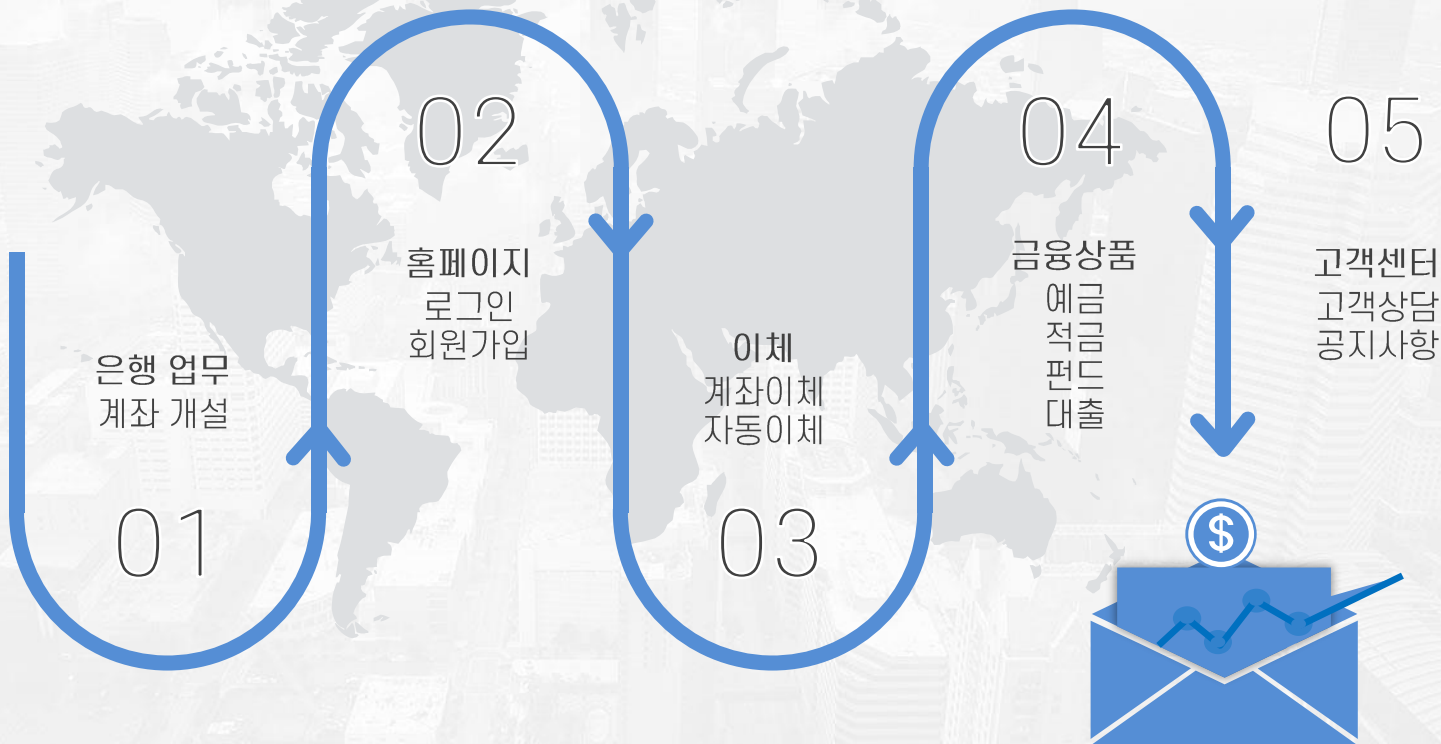
- ✓ 데이터와 기능의 집합을 제공하여 어떤 자원에 대해 CRUD 연산을 수행하기 위해 URI로 요청을 보내는 것





특정한 시간에 원하는 일을
자동으로 시키는 것.
JOB으로 사용될 서비스 메소드에
@Scheduler(cron 설정) 후 이용

업무흐름도 >>>



Schedule

: 프로젝트 일정

개인별 프로젝트 일정

김은진

[illegible]

[illegible]

최슬기

[illegible]

한채영

[illegible]

[illegible]

[illegible]

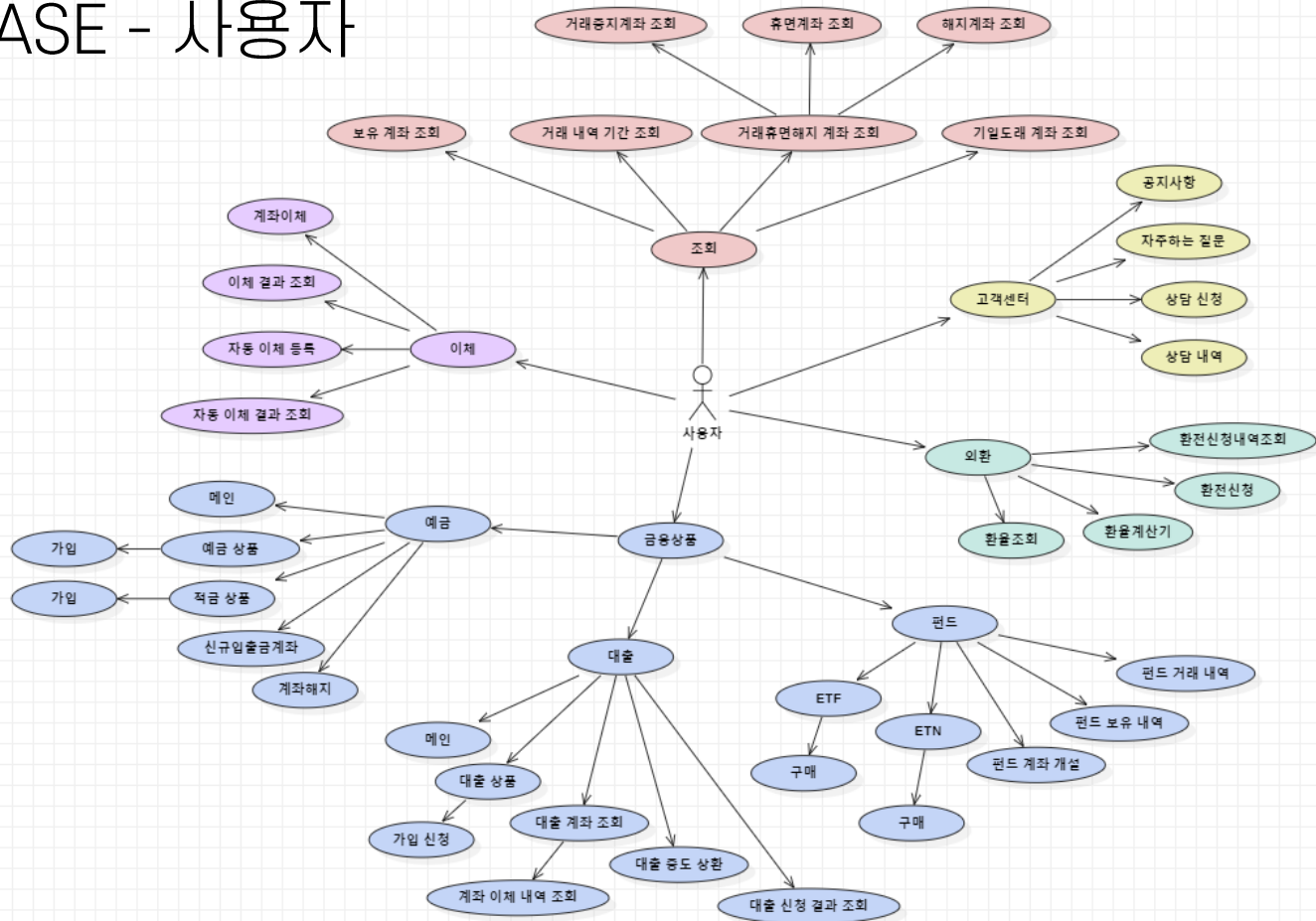


Design

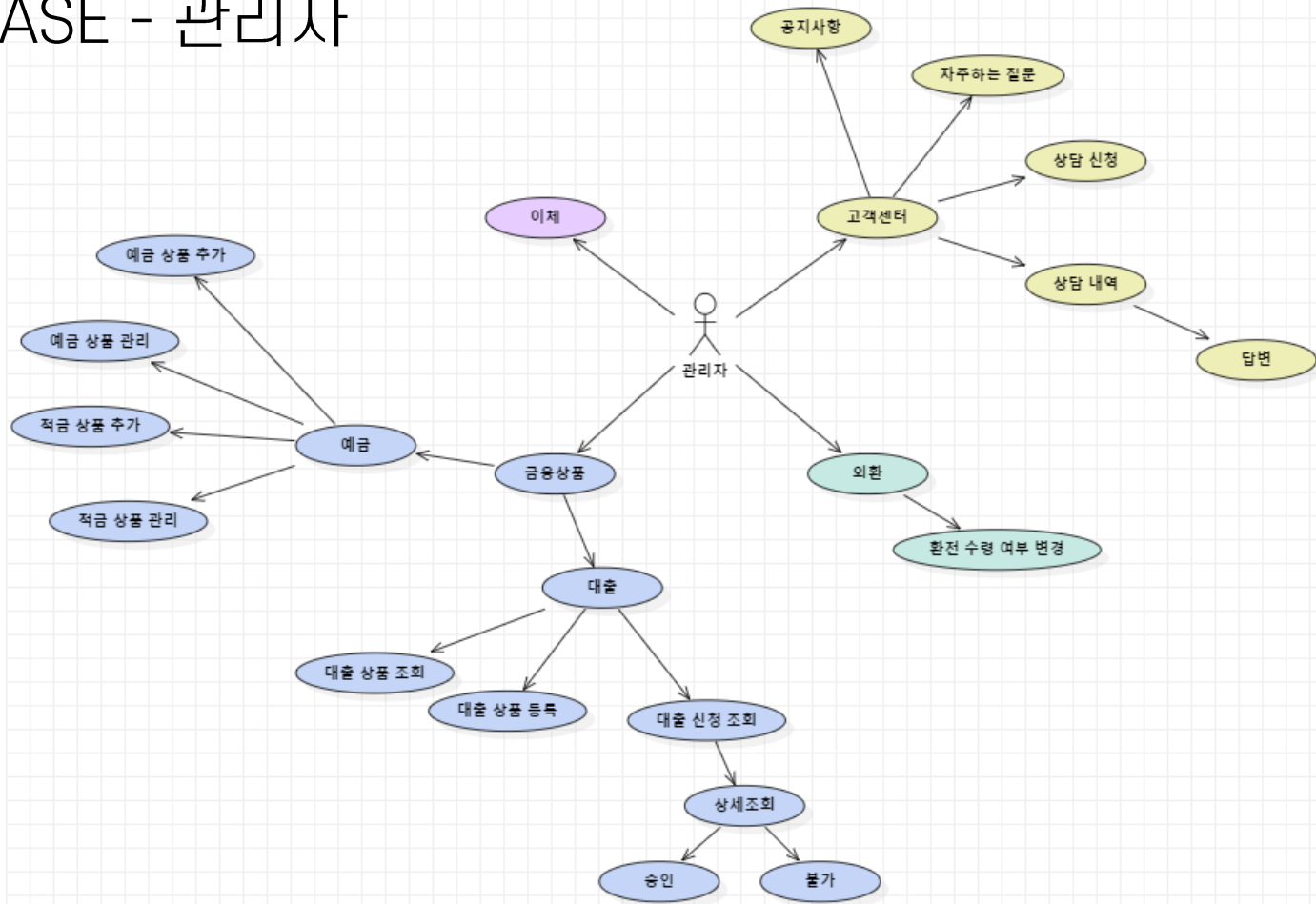
: 프로젝트 설계

UseCaseDiagram, ERD, 기능정의서, 주요메뉴

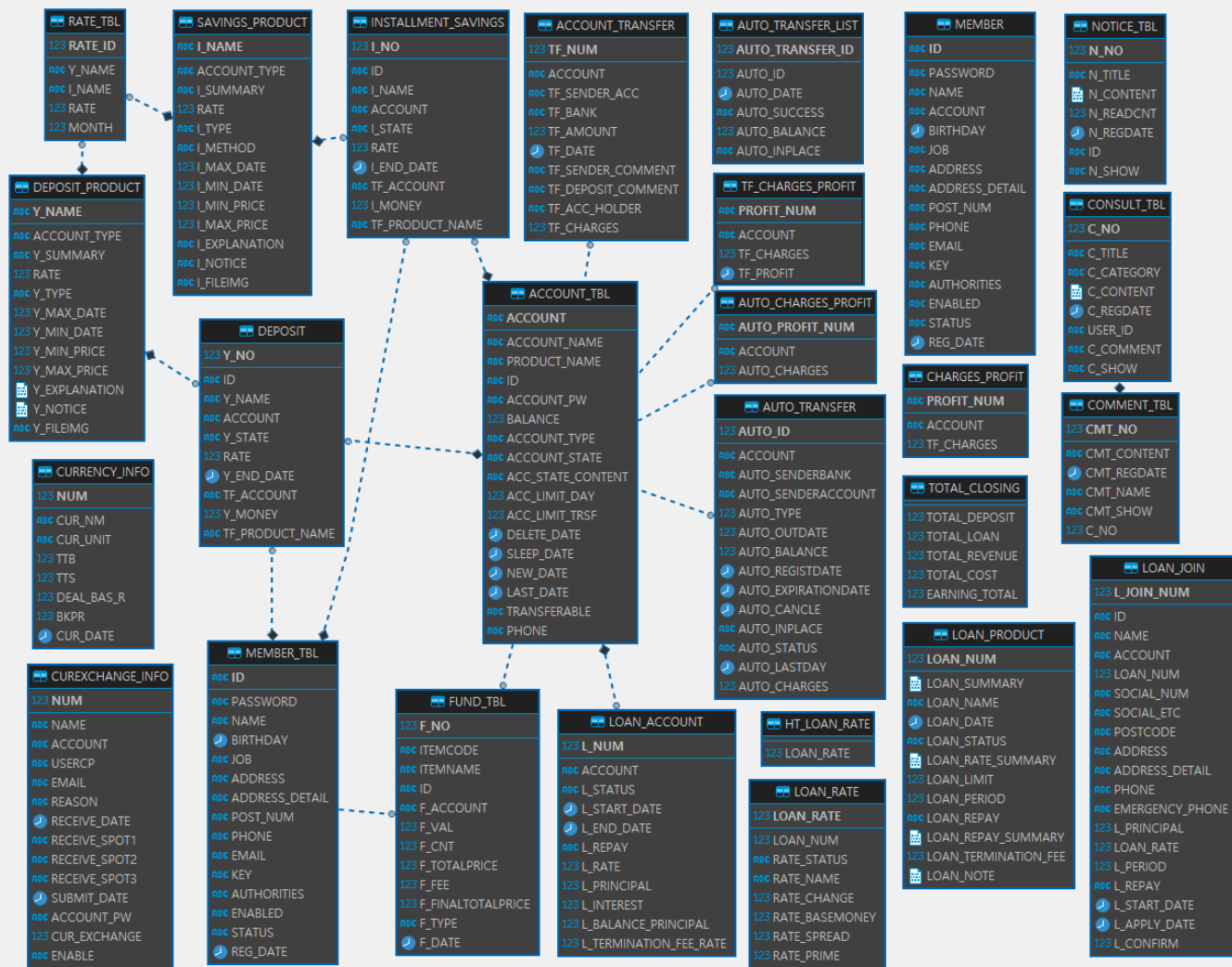
USECASE - 사용자



USECASE - 관리자



ERD



기능정의서 - 사용자

1차	2차	3차	4차	구현 기능	메뉴 출력
사용자	회원가입		회원가입	시큐리티, bcrypt 비밀번호 암호화	시큐리티, bcrypt 암호화
			본인 인증	이메일 인증번호 확인기능	시큐리티, bcrypt 암호화
	로그인			시큐리티, bcrypt 비밀번호 암호화	시큐리티, bcrypt 암호화
	계좌정보		계좌 상태 변경신청	휴면 계좌 정상화 및 계좌 해지 신청	ajax
	이체	계좌이체	잔액 조회	계좌 이체 잔액 조회 서비스	스케줄러
			한도 조회	계좌 이체 한도 조회 서비스	
			계좌 이체	계좌 이체 서비스	
		자동이체	조회/해지	자동이체 여부 조회/해지 서비스	
			자동이체 신청	스케줄러를 이용하여 자동이체 신청	
		한도변경	한도 조회	일일 한도 조회	
			한도 변경 신청	일일 한도 설정 및 변경 서비스	
	조회이체	계좌등록	계좌 개설	신규 계좌 개설 서비스	Vue
		계좌조회	보유계좌조회	보유중인 계좌 조회 (한도제한계좌 포함)	
			거래내역조회	최근 및 기간별 거래내역 조회	
			휴면 및 해지계좌조회	휴면 및 해지 상태의 계좌 조회	
			기일도래계좌조회	대출 만기도래현황 조회(기일경과 혹은 1개월이내 도래분)	
	외환	환율	환율정보	오픈 api를 이용한 환율 정보 제공	api, ajax api, ajax
			환율계산기	환율 계산 서비스	
			환전신청	USD 환전 신청 및 수령일, 지점 예약 서비스	
			환전신청내역조회	환전 신청한 내역 조회	
			대출 계좌 조회	보유 중인 대출 상품 조회	
	대출	대출관리	대출 계좌 거래내역	대출 계좌 거래내역	ajax
			대출 중도 상환	대출 중도 상환	
			대출 해지 현황 조회	해지된 대출 상품 조회	
			대출 해지 현황 상세 조회	해지된 대출 정보 / 거래 내역	
			원금 균등 분할 상환	원금 분할 납부 방식	
		대출신청	원리금 균등 분할 상환	원금+이자 분할 납부 방식	
			원금 만기일시 상환	매달 이자 상환 / 만기일시 원금 상환	

기능정의서 - 사용자

1차	2차	3차	4차	구현 기능	메뉴 출력
사용자	금융상품	예금	예금 상품 조회	은행 예금상품 조회&이자계산기	vue, ajax
			예금 상품 신청	예금 상품 가입	
			보유 예금 조회	가입한 예금 조회	
			예금 납부	예금 납부(계좌이체, 자동이체)	
		적금	예금 해지	예금 중도 해지	
			적금 상품 조회	적금 목록 조회&이자계산기	vue, ajax
			적금 상품 신청	적금 가입	
			가입 상품 조회	가입한 상품 조회	
	펀드	펀드조회	적금 납부	적금 납부(계좌이체, 자동이체)	
			적금 해지	적금 중도 해지	
			펀드리스트	펀드리스트	네이버 파이낸스 api
			상세페이지	상세페이지	네이버 파이낸스 api
		펀드 거래	계좌개설	계좌개설	
			펀드 구매	펀드 구매	네이버 파이낸스 api
			펀드 판매	펀드 판매	네이버 파이낸스 api
		펀드 내역	펀드 보유 내역	펀드 보유 내역 조회	네이버 파이낸스 api
			펀드 거래 내역	펀드 거래 내역 조회	네이버 파이낸스 api
	고객센터	고객상담	상담게시판	상담글 목록 조회	
				상담글 상세 조회	ajax
		공지사항	공지게시판	상담글 입력, 수정, 삭제	Summernote api
				공지사항 목록 조회	
				공지사항 상세 조회	

기능정의서 - 관리자

1차	2차	3차	4차	구현 기능	메뉴 출력
관리자	고객관리	회원정보	회원 정보 수정	회원정보 수정	
		계좌정보	계좌 비밀번호 변경	계좌 비밀번호 변경	
			계좌 상태 변경	해지, 휴면 및 정지처리	ajax, chart.js
			주제별 계좌 검색	계좌 / 이름 / 종류 / 아이디 검색	ajax, chart.js
	고객관리		이체 내역 조회	거래번호 / 계좌 / 아이디 / 예금주 / 입금자 별 이체 내역 조회	
		이체	계좌 이체 내역 조회	계좌 이체 내역 조회	
		자동이체	조회	자동이체 여부 조회 서비스	
	외환	환율	환전신청내역 수령확인	환전신청내역 수령여부 확인하여 변경	
	금융상품	예금 상품	조회	예금 상품 조회	
			등록	예금 상품 등록	
			수정	예금 상품 수정	
			삭제	예금 상품 삭제	
			이자계산	중도해지 이자 자동 지급	
		적금 상품	이자계산	만기시 해지 자동 이자 지급	스케줄러
			조회	적금 상품 조회	
			등록	적금 상품 등록	
			수정	적금 상품 수정	
			삭제	적금 상품 삭제	
	대출	대출 상품 관리	이자계산	중도해지 이자 자동 지급	
			이자계산	만기시 자동 이자 지급	스케줄러
		대출 신청 관리	조회	대출 상품 조회	Vue + JPA
			등록	대출 상품 등록	Vue + JPA
			대출 신청 조회	대출 신청 목록 및 상세 조회	
			대출 신청 처리	대출 신청 승인 및 불가 처리	
			대출 계좌 생성	대출 승인 시 자동 생성	
			대출 상환 자동 이체 생성	대출 승인시 자동 생성	

기능정의서 - Vue관리자

1차	2차	3차	4차	구현 기능	메뉴 출력
Vue	고객관리	계좌정보	예금주명 조회	예금주명으로 조회	Vue + JPA
			계좌번호 조회	계좌번호로 조회	
		이체	계좌 이체 내역 조회	계좌 이체 내역 조회	
		자동이체	조회	자동이체 여부 조회 서비스	
		공지사항	조회	공지사항 조회	
				공지사항 상세 조회	
	대출	대출 상품 관리	조회	대출 상품 조회	
			등록	대출 상품 등록	

주요메뉴

계좌관리 및 조회

- Ajax
- Chart.js

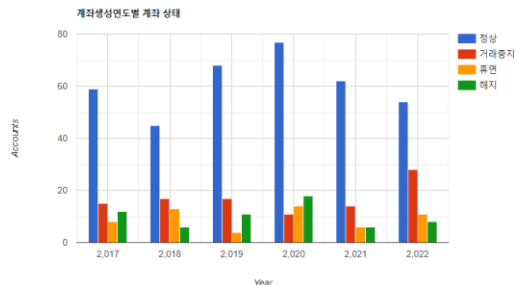
JQuery Ajax를 사용해서
고객의 계좌를 조회하거나
상태를 변경하고,
차트API를 이용해 현황을
출력하는 코드

화면

계좌 관리

- > 계좌정보조회
- > 계좌상태 변경
- > 입출금 내역조회
- > 결산내역 조회

종합정보	예금주명	계좌번호	회원ID	계좌종류	기일도래계좌	계좌상태
총고객수	가입ID	통계개좌	유연계좌	당월신규	당월미유역	총잔액(단위:만원)
584	180	584	56	18	17	562509



종합정보	예금주명	계좌번호	회원ID	계좌종류	기일도래계좌	계좌상태	
예금주	ID	계좌번호	계좌종류	신규발	원리금계좌 (기일도래)	잔액	계좌상태
test	han	898023142802	직공	2022-10-23	2022-10-23	0	정상
예금주 314	han	555000000314	입출금	2021-11-15	2022-10-11	4465285	거래중지
test	han	205498796416	입출금	2022-10-18	2022-10-18	1105000000	휴면
예금주 315	han	555000000315	입출금	2019-12-12	2022-10-08	1566292	거래중지
예금주 319	han	555000000319	입출금	2018-10-10	2022-10-01	3221238	거래중지
test	han	596700081207	대출	2022-10-19	2022-10-24	0	정상
test	han	735538423378	환도	2022-10-18	2022-10-18	53440	휴면
통도통	han	000000001111	입출금	2022-10-19	2022-10-24	0	정상
예금주 310	han	555000000310	입출금	2019-12-18	2022-10-04	1183985	거래중지
test	han	673398802032	대출	2022-10-21	2022-10-23	0	해지

조회

```

<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
<script type="text/javascript">
    $(function(){
        $('#input[name="account_btn"]').click(function(){
            var param = {
                "searchType": $(this).next().val(),
                "searchValue": $(this).prev().val(),
            }
            $.ajax({
                url: '${path}/adminSearchAction.ad?${_csrf.parameterName}=${_csrf.token}'
                type: 'post',
                data: param,
                success: function(data){
                    $('#content1').html(data);
                },
                error: function(){
                    alert('알지하는 정보가 없습니다. 다시 확인해주세요!');
                }
            });
        });
    });

    google.charts.load('current', {'packages':['corechart']});
    google.charts.setOnLoadCallback(drawVisualization);

    function drawVisualization() {
        var data = google.visualization.arrayToDataTable([
            ['year', '정상', '거래중지', '휴면', '해지'],
            ${r}
        ]);
        var options = {
            title: '계좌생성연도별 계좌 상태',
            vAxis: {title: 'Accounts'},
            hAxis: {title: 'Year'},
            seriesType: 'bars',
            series: {type: 'line'}}
        };
        var chart = new google.visualization.ComboChart(document.getElementById('chart_div'));
        chart.draw(data, options);
    }
</script>

<!-- 연도별+상태별 계좌 현황 : 관리자페이지 Chart -->
<select id="cntAccountYearState" resultType="com.mvc.upbank.dto.AdminSearchDTO">
    select count(CASE WHEN account_state='정상' THEN 1 END) AS cnt_account_normal
    , count(CASE WHEN account_state='거래중지' THEN 1 END) AS cnt_account_stopped
    , count(CASE WHEN account_state='휴면' THEN 1 END) AS cnt_account_sleep
    , count(CASE WHEN account_state='해지' THEN 1 END) AS cnt_account_cancelled
    from account_tbl
    group by substr(new_date,1,2)
</select>

```

주요메뉴

자동이체

- Scheduler

스케줄러를 이용하여
매일 같은 시간에 자동으로
계좌에서 금액을 출금하는 코드

화면

자동이체결과조회

이체기관	출금 계좌	입금 계좌	금액	주기	시작일	종료일	종류	상태	최근납부	해지일
대출이자	372720941774	110384123151 2	300,000원	매달 7일	2022-10-24	2026-10-23	대출	정상	내역없음	해지
정기예금	372720941774	333992929188 2	50,000원	매달 15일	2022-10-10	2025-10-09	예금	정상	2022-10-15	해지
신한정기적금	372720941774	110384121541	200,000원	매달 10일	2022-09-24	2025-09-23	적금	해지	2022-10-10	2022-10-24

```
// 내 계좌 잔액조회
account_balance = dao.selectAccountBalance(account);

if (account_balance >= auto_balance) {
    // 이체시작
    // 1. 이체데이터물에 내역 추가
    TransferDTO dto = new TransferDTO();
    dto.setACCOUNT(account);
    dto.setTf_sender_acc(auto_senderAccount);
    dto.setTf_amount(auto_balance);
    dto.setTf_sender_comment(auto_inPlace);
    dao.insertTranferByAuto(dto);

    // 2. 자동이체내역데이터물에 내역추가
    Map<String, Object> map = new HashMap<String, Object>();
    map.put("auto_id", auto_id);
    map.put("id", id);
    map.put("auto_balance", auto_balance);
    map.put("auto_inplace", auto_inPlace);
    dao.insertAutoTransferList(map);

    // 3. 납부한거 계좌반영(내 계좌 잔액감소)
    Map<String, Object> map1 = new HashMap<String, Object>();
    map1.put("account_balance", (account_balance - auto_balance));
    map1.put("account", account);
    dao.updateAccountAutoTransfer(map1);

    // 자동이체 데이터물에 최신납부내역 갱신
    Logger.info("자동이체 성공");
    // 4. 자동이체 데이터물에 납부내역 갱신
    dao.updateAutoTransfer(auto_id);
} else {
    Logger.info("자동이체 실패 : 계좌잔액이 부족합니다.");
    // 자동이체내역리스트에 실패내역 추가
    dao.failAutoTransferList(auto_id);
}

}

{
    Logger.info("자동이체할 데이터가 없습니다.");
}
```


주요메뉴

예/적금 이자 지급 이자 계산기

Ajax를 활용해 만든
이자 수익 계산기와
해지 시 이자 수익을
자동 지급하는 코드

화면

예금계산기

5,000,000 을 24 개월 간 5 %의 예금상품에 저축하면? [결과보기](#) [조기화](#)

[+0.1%](#) [+1%](#) [+5%](#)

500,000원을 더해 총 5,500,000원을 모으실 수 있습니다.

구분	원금금액	이자
비과세	5,500,000원	500,000원
일반과세 (세후)	5,423,000원	423,000원

- 일반과세의 경우 이자금액의 15.4%가 원천징수되고 비과세 종합저축의 경우 이자소득세가 면제됩니다.
- 비과세종합저축은 가입대상자에 한해 5천만원 한도로 적용됩니다.
- 관련세법에 따른 세율변경 시 변경된 세율이 적용됩니다.
- 본 계산결과는 단기 및 이자금액 계산을 위한 단순 예시로 각 상품별 세제혜택 내용에 따라 달라질 수 있습니다.

UP Star 정기예금

정보입력

가입기간 1~36개월, 월단위 개월 12개월

가입금액 최소100만원 이상, 원단위 원 1000만 500만 300만 100만

입금종 계좌 확인 upBank 기본 입출금통장 - 372720941774

출금가능금액 출금가능금액: 85,412,500원

계좌 비밀번호 숫자 4자리를 입력하세요

신규 비밀번호 숫자 4자리를 입력하세요

비밀번호 확인 숫자 4자리를 입력하세요

[가입하기](#) [조기화](#) [목록](#)

```
Map<String, Object> map = new HashMap<String, Object>();
int[] month = { 1, 6, 12, 24, 36 };
for (int i = 0; i < month.length; i++) {
    System.out.println("rate" + month[i] + " : " + req.getParameter("rate" + month[i]));

    try {
        map.put("rate", Double.parseDouble(req.getParameter("rate" + month[i])));
        dto.setRate(Double.parseDouble(req.getParameter("rate" + month[i])));
        updateCnt = dao.depositUpdate(dto);
    } catch (Exception e) {
        e.printStackTrace();
        try {
            dto.setRate(Double.parseDouble(req.getParameter("rate" + month[i - 1])));
            updateCnt = dao.depositUpdate(dto);
            map.put("rate", 0);
        } catch (Exception e1) {
            e1.printStackTrace();
        }
    }
}
```

```
Map<String, Object> map2 = new HashMap<String, Object>();
map2.put("ACCOUNT", ACCOUNT);
map2.put("id", id);
DepositDTO dto = dao.depositCancel(map);
SearchDTO dto2 = dao2.SearchAccountColumn(map2);
Map<String, Object> map5 = new HashMap<String, Object>();
map5.put("ACCOUNT", dto.getTf_account());
map5.put("id", id);
SearchDTO dto3 = dao2.SearchAccountColumn(map5);
double oldBal = dto3.getBALANCE();
double getVal = dto2.getBALANCE();
double getRate = dto.getRate() / 100;
double Balance = oldBal + getVal + getVal * getRate;
int Bal = (int) Balance;
model.addAttribute("Bal", Bal);
model.addAttribute("getRate", getRate);
Map<String, Object> map3 = new HashMap<String, Object>();
map3.put("id", id);
map3.put("BALANCE", Bal);
map3.put("PRODUCT_NAME", dto.getTf_product_name());
int updateCnt2 = dao.depositMoneyPlus(map3);
if (updateCnt2 == 1) {
    Map<String, Object> map4 = new HashMap<String, Object>();
    map4.put("id", id);
    map4.put("PRODUCT_NAME", y_name);
    int updateCnt3 = dao.depositMoneyMinus(map4);
    model.addAttribute("updateCnt3", updateCnt3);
}
```

주요메뉴

대출 이자 및 상환액 대출 계산기

수식을 적용하여 대출시 상환해야할 금액과 총 이자액을 출력하는 코드

화면

대출 메인



대출 상품 찾기



대출 진행 조회



대출금 상환

대출 계산기

원리금 균등 상환

원금 균등 상환

원금 만기일시 상환

원리금 균등 상환일 때,

대출 금액 만원을 기간 년 동안
이자 %로 대출 받으려면?

결과보기

대출 계산기 : 원리금 균등 상환

회차	이자	원금	내야하는 금액
1	26,667원	821,181 원	847,848 원
2	24,477원	823,371 원	847,848 원
3	22,281원	825,567 원	847,848 원
4	20,080원	827,768 원	847,848 원
5	17,872원	829,976 원	847,848 원
6	15,659원	832,189 원	847,848 원
7	13,440원	834,408 원	847,848 원
8	11,215원	836,633 원	847,848 원
9	8,984원	838,864 원	847,848 원
10	6,747원	841,101 원	847,848 원
11	4,504원	843,344 원	847,848 원
12	2,255원	845,593 원	847,848 원
총 이자액			174,181 원

```
// 대출 이자 계산기 -----
// 대출 이자 계산기 - 원리금 균등 분할 상환
public void amortization(HttpServletRequest req, Model model) {
    double principal = Integer.parseInt(req.getParameter("amortization_principal")) * 10000; //
    double rate = Double.parseDouble(req.getParameter("amortization_interest")); // 대출 이율
    int month = Integer.parseInt(req.getParameter("amortization_year")) * 12; // 대출 기간

    double r = rate * 0.01 / 12; // 월 이자율 = 이자율 * 0.01 / 개월수

    double rn = Math.pow(1+r, month); // 1+R의 (개월수) 거듭제곱
    double son = principal * r * rn; // 분자
    double par = rn - 1; // 분모
    int installment = (int) Math.round(son / par); // 매회 상환해야하는 금액

    double interest = 0; // 총 이자
    int balance = (int) principal; // 잔액
    int total_interest = 0; // 납부해야할 이자 총액

    // 값을 담은 리스트 생성
    ArrayList<Map<String, Object>> list = new ArrayList<Map<String, Object>>();

    for(int i=1; i<=month; i++) {
        // 리스트에 담길 해쉬맵 생성
        Map<String, Object> map = new HashMap<String, Object>();

        interest = Math.round(balance * r);
```

```
int repay = (int) Math.round(installment - interest);
balance -= repay;
total_interest += interest;

// 맵에 해당값을 넣기
map.put("count", i); // 회차
map.put("installment_principal", repay); // 각 회차당 납부
map.put("installment_interest", interest); // 각 회차당 납부
map.put("installment", installment); // 각 회차당 납부액

// 리스트에 맵 넣기
list.add(map);
}

model.addAttribute("list", list);
model.addAttribute("total_interest", total_interest); // 납
```

주요메뉴

대출 상품 조회

- Restful API
 - JPA
 - Vue

JPA와 Vue를 이용한 데이터 조회

화면

상품 목록

UP 직장인든든 신용대출

직장인이려면

최고 **3억** 원

UP STAR CLUB 신용대출

UP STAR CLUB 고객을 위한 우수고객 전용 상품

최고 **2억** 원

UP 사업자든든 신용대출

자영업자 및 프리랜서 고객님들의 생활안정을 위한

최고 **1억** 원

상품&가입

직장인이려면
UP 직장인든든 신용대출



기간
최장 10 년



상환 방법
원리금균등분할상환



최고
30 천만원

상품안내 금리 및 이율 이용 안내 유의사항 및 기타

대출 한도

→ 최대 3억원

대출기간 및 상환
방법

→ 대출기간: 최장 10년 이내
→ 상환 방식: 원리금균등분할상환

원리금상환방법

→ 원리금 이월할 경우 상환금액이 늘어 납니다. 원리금 이월하지 않고, 원리금 상환금액과 원리금 이월금 두 부분

목록

```
import axios from "axios"

export default {
  namespaced: true,
  state: {
    productDetail: {}
  },
  mutations: {
    setProductDetail(state, payload) {
      state.productDetail = payload
    },
    clearProductDetail(state) {
      state.productDetail.length = 0
    }
  },
  actions: {
    async fetchDetail({ state, commit }, payload) {
      commit('clearProductDetail')

      try {
        let response = await axios.get("http://localhost:8040/api/product_detail?loan_num=" + payload, {
          headers: {
            'Content-Type': 'application/json; charset=utf-8',
          }
        })

        let temp = {}
        for(const [key, value] of Object.entries(response.data.detail)) {
          temp[key] = value ?? ''
        }
        commit('setProductDetail', temp)
      } catch(error) {
        console.log(error)
      }
    }
  }
}
```

```
@Entity(name="loan_product")
public class LoanProductVO {
    @Id @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int loan_num; // 대출 상품 번호

    private String loan_name; // 대출 상품 이름
    private String loan_summary; // 대출 상품 요약
    private Date loan_date; // 대출 상품 등록일
    private String loan_status; // 상태 - 판매중, 판매중단
    private String loan_rate_summary; // 금리 설명
    private double loan_limit; // 대출 한도
    private String loan_period; // 대출 기간
    private String loan_repay; // 상환 방식
    private String loan_repay_summary; // 상환 방식 설명
    private double loan_termination_fee; // 중도상환 해약금
    private String loan_note; // 유의사항

    @OneToMany(fetch=FetchType.EAGER)
    @JoinColumn(name="loan_num")
    private List<LoanProductRateVo> rate = new ArrayList<LoanProductRateVo>();
}
```

주요메뉴

펀드 목록 펀드 상세조회

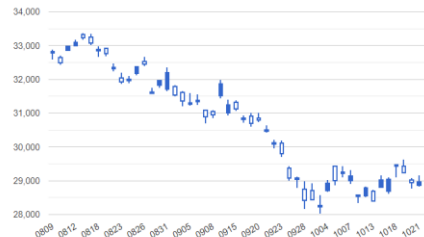
- 네이버 파이낸스 api
- 구글 차트 api

Json, xml 데이터를 파싱해서
실제 데이터를 가져와
차트형태로 펀드를 구성하여
화면에 출력하는 코드

화면

ETF

종목코드	종목명	현재가	전일비	등락률	NAV	시가총액(백만원)	거래량	거래대금(백만원)	시가총액(억)
069500	KODEX 200	29,255	△270	△0.93	29,326	9,936,724	144,975	51,942	
377460	TIGER 미국나스닥100	13,625	▼155	▼1.12	13,664	17,49	3,495,886	46,016	32,259
423160	KODEX KOSPI200	101,080	△15	△0.01	101,079	△0.65	83,997	8,490	30,201
252670	KODEX 200선물인버스2X	3,995	▼70	▼1.91	3,974	△19.58	164,278,275	585,458	22,817
133690	TIGER 미국나스닥100	73,215	△2,525	△3.57	72,854	▼2.76	191,478	13,990	21,481
122630	KODEX 미국채	12,625	△240	△1.94	12,732	▼19.42	34,476,891	438,768	21,374
360750	TIGER 미국S&P500	13,590	△375	△2.84	13,574	△1.61	1,391,077	18,893	16,747
100210	TIGER 200	29,280	△280	△0.97	29,346	▼9.38	1,073,870	31,522	18,329
278530	KODEX 200TR	9,775	△100	△1.03	9,769	▼9.41	350,540	3,424	17,830
100280	KODEX 삼성그룹	8,525	△185	△2.22	8,554	▼1.42	435,567	3,895	16,504
292150	TIGER TOP10	9,560	△65	△0.68	9,553	▼11.22	118,660	1,137	15,846
357870	TIGER KOSPI200	51,325	△0	△0.00	51,326	△0.67	197,353	10,129	15,573
157450	TIGER 단기물인버스	101,485	△5	△0.00	101,489	△0.48	607,119	61,615	15,291
214980	KODEX 단기채	104,165	▼25	▼0.02	104,172	△0.44	202,290	21,072	14,835
381170	TIGER 미국채	10,575	△390	△3.83	10,537	▼4.95	483,448	5,106	14,234
381180	TIGER 미국채	9,060	△390	△4.50	9,040	▼13.77	1,592,712	14,406	14,011
305720	KODEX 2차전지	20,180	△390	△1.97	20,246	△9.70	1,468,095	29,488	13,319
305540	TIGER 2차전지	18,950	△365	△1.96	18,980	△2.12	1,218,639	23,038	12,734



```
String line = "";
try{
```

```
String address = "https://finance.naver.com/api/sise/etnItemList.nhn"; //네이버 펀드 파이낸스
```

```
URL url = new URL(address);
URLConnection urlConn = url.openConnection();
urlConn.setRequestProperty("Content-Type", "application/json;UTF-8");
```

```
InputStreamReader ir = new InputStreamReader(urlConn.getInputStream(), "EUC_KR");
BufferedReader br = new BufferedReader(ir);
```

```
line = br.readLine();
```

```
JSONParser parser = new JSONParser();
//데이터를 파싱하여 배열로 추출
Object obj = parser.parse(line);
```

```
JSONObject jsonMain = (JSONObject)obj;
JSONObject jsonSub = (JSONObject)jsonMain.get("result");
JSONArray jsonArr2 = (JSONArray)jsonSub.get("etnItemList");
if(jsonArr2.size() > 0){
    List<FundDTO> list = new ArrayList<>();
```

```
for(int i = 0 ; i < 100 ; i++){
    FundDTO dto = new FundDTO();
    JSONObject jsonObj = (JSONObject)jsonArr2.get(i);

    //아이템 코드
    String itemcode = (String) jsonObj.get("itemcode");
    dto.setItemcode(itemcode);

    //거래대금
    String amount_ = String.valueOf(jsonObj.get("accAmount"));
    int amonut = Integer.parseInt(amount_);
    dto.setAmonut(amonut);

    String itemname = String.valueOf(jsonObj.get("itemname"));
    String nowVal_ = String.valueOf(jsonObj.get("nowVal"));
    int nowVal = Integer.parseInt(nowVal_);
    String changeVal_ = String.valueOf(jsonObj.get("changeVal"));
    float changeVal = Float.parseFloat(changeVal_);
    String changeRate_ = String.valueOf(jsonObj.get("changeRate"));
    float changeRate = Float.parseFloat(changeRate_);
    String quant_ = String.valueOf(jsonObj.get("accQuant"));
    int quant = Integer.parseInt(quant_);
    String marketSum_ = String.valueOf(jsonObj.get("marketSum"));
    int marketSum = Integer.parseInt(marketSum_);

    dto.setChangeRate(changeRate);
    dto.setChangeVal(changeVal);
    dto.setItemname(itemname);
    dto.setMarketSum(marketSum);
    dto.setNowVal(nowVal);
    dto.setQuant(quant);
    list.add(dto);
}

model.addAttribute("list", list);
}
```

계좌정보 조회
환율정보 및 환전신청

환율정보 제공 및,
환율이 제공되지 않는
공휴일 표기를 위해
JSON 및 XML 타입의 API를
파싱 하여 환율 및 공휴일 정보를
가져오는 코드

외환

On

화을 조희

기준종	종가(원)				종가(달러)
종목명	종가(원)	종가(달러)	종가(원)	종가(달러)	종가(달러)
미국표준지하의 다우존스	AED	381.2	395.03	391.12	391
코스닥지	AUD	905.33	623.63	914.47	914
미국표준지하의 다우존스	BID	3777.7	3848.41	3810.31	3810
미국표준지하의 다우존스	END	3103.33	1823.5	1823.47	1823
미국표준지하의 다우존스	END	1041.69	1162.4	1162.22	1162
미국표준지하의 다우존스	CHAI	1426.95	1488.66	1435.31	1435
미국표준지하의 다우존스	CHAI	105.79	104	104.77	105
미국표준지하의 다우존스	ENK	338.18	311.96	306.69	306
미국표준지하의 다우존스	ENK	1395.9	1428.19	1414.85	1414
미국표준지하의 다우존스	GGP	1105.47	1141.98	1126.73	1126
미국표준지하의 다우존스	HSD	1878.18	1884.85	1853.62	1853

환전 신청 - 1

통화종류	원천금액(외화)	현재고시원율	무대석원율	무대율	원천금액(원)
USD	1000	1426.6	1431.6	50%	1,431,600원

한정신형

계좌조회

계좌명	계좌번호	신규일	최근거래일 (7년기)	잔액	유형	업무
upBank 기본 입출금통장	616730202781	2022-10-19	2016-12-11	8,909,639원	입출금	조회 이체
UP STAR CLUB 신용대출	928853626896	2022-10-24	2022-10-24	-2,020,150원	대출	조회
upBank 자유로움 입출금통장	441727146987	2022-10-19	2016-12-11	500,000,000원	입출금	조회 이체
UP 직장인도도 신용대출	891806483980	2022-10-19	2016-12-11	-3,156,024원	대출	조회

```
String line = "";
try{
```

```
String address = "https://finance.naver.com/api/sise/etnItemList.nhn"; //네이버 펀드 파이낸스
```

```
URL url = new URL(address);
URLConnection urlConn = url.openConnection();
urlConn.setRequestProperty("Content-Type", "application/json;UTF-8");
```

```
InputStreamReader ir = new InputStreamReader(urlConn.getInputStream(), "EUC_KR");
BufferedReader br = new BufferedReader(ir);
```

```
line = br.readLine();
```

```
JSONParser parser = new JSONParser();
//데이터를 파싱하여 배열로 추출
Object obj = parser.parse(line);
```

```
JSONObject jsonMain = (JSONObject)obj;
JSONObject jsonSub = (JSONObject)jsonMain.get("result");
JSONArray jsonArr2 = (JSONArray)jsonSub.get("etnItemList");
if(jsonArr2.size() > 0){
    List<FundDTO> list = new ArrayList<>();
```

```
for(int i = 0 ; i < 100 ; i++){
    FundDTO dto = new FundDTO();
    JSONObject jobj = (JSONObject)jsonArr2.get(i);

    //아이템 코드
    String itemcode = (String) jobj.get("itemcode");
    dto.setItemcode(itemcode);

    //거래대금
    String amount_ = String.valueOf(jobj.get("accAmount"));
    int amonut = Integer.parseInt(amount_);
    dto.setAmonut(amonut);

    String itemname = String.valueOf(jobj.get("itemname"));
    String nowVal_ = String.valueOf(jobj.get("nowVal"));
    int nowVal = Integer.parseInt(nowVal_);
    String changeVal = String.valueOf(jobj.get("changeVal"));
    float changeVal = Float.parseFloat(changeVal_);
    String changeRate_ = String.valueOf(jobj.get("changeRate"));
    float changeRate = Float.parseFloat(changeRate_);
    String quant_ = String.valueOf(jobj.get("accQuant"));
    int quant = Integer.parseInt(quant_);
    String marketSum_ = String.valueOf(jobj.get("marketSum"));
    int marketSum = Integer.parseInt(marketSum_);

    dto.setChangeRate(changeRate);
    dto.setChangeVal(changeVal);
    dto.setItemname(itemname);
    dto.setMarketSum(marketSum);
    dto.setNowVal(nowVal);
    dto.setQuant(quant);
    list.add(dto);
}

model.addAttribute("list", list);
}
```



Action : 프로젝트 시연

UP Bank 시연

UpBank

조회 이체 금융상품 외환 고객센터

유정리와 함께 부자합시다!

#부자되는 법이 궁금하면?
#비법틀고 커피 한잔!



조회

이체

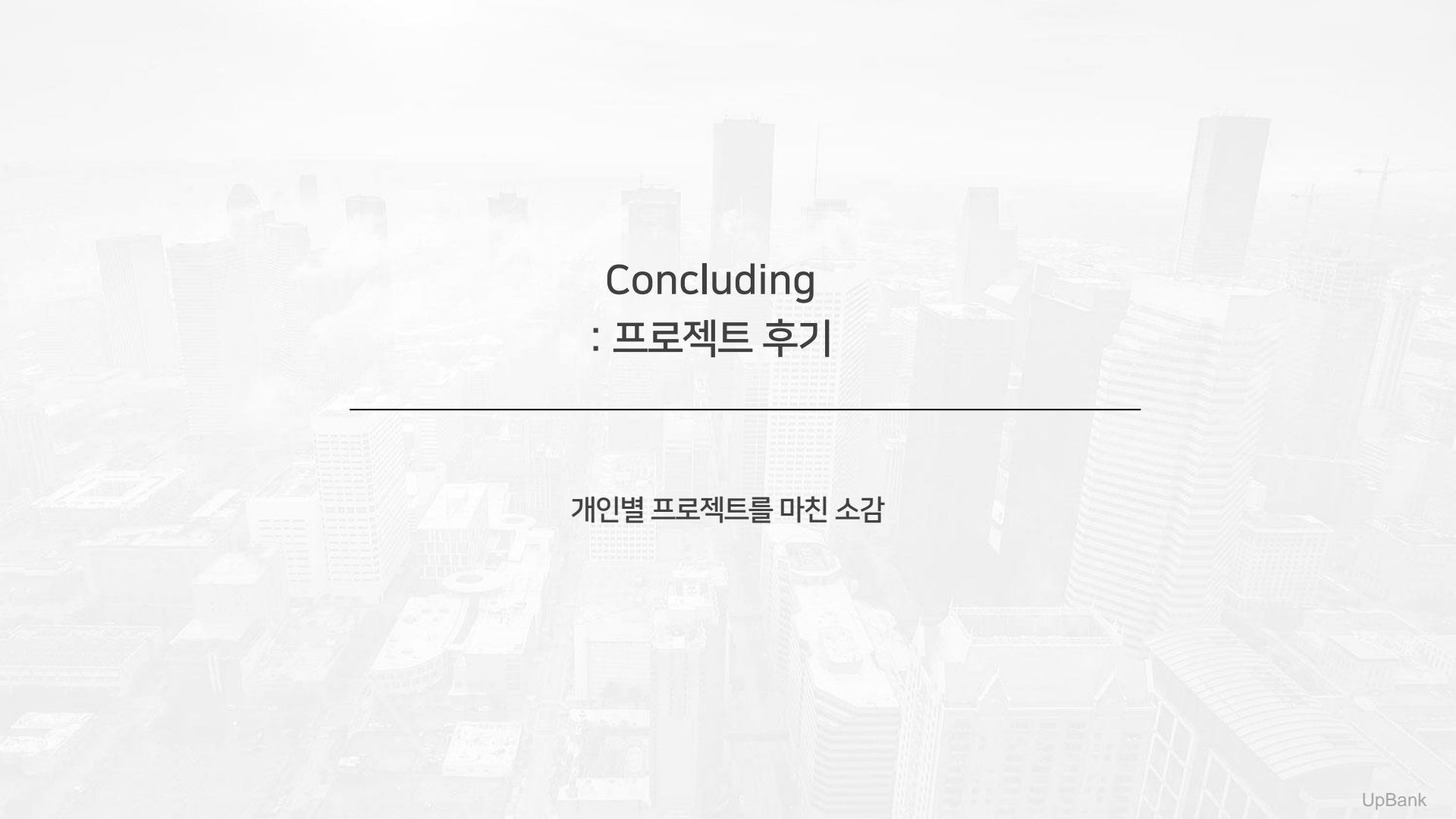


예적금
부자되는 알짜정보

펀드
한 눈에 보는 펀드

환전지갑
알뜰하게 환전하기

대출
UpBank 든든대출



Concluding : 프로젝트 후기

개인별 프로젝트를 마친 소감

프로젝트 후기

Concluding

김은진

프로젝트 진입에 앞서 선행되는 분석 설계의 중요성을 느꼈습니다. 팀원들의 각기 다른 장점이 모여 하나의 결과물을 완성해내는 것을 보며 협업의 시너지효과를 체감하였고, 자유롭고 수평적인 의사소통의 중요성을 다시 한 번 깨닫게 되었습니다.

김아람

낮선 업무를 맡아 초반 설계에서 혼란을 겪었습니다. 구축하는 중에 많은 데이터를 수정하며 기초 설계가 얼마나 중요한지 깨달았습니다. 평소 알고 지내던 사이라도 함께 일하는 건 쉽지 않았습니다. 손발을 맞춰가는 것에도 기술이 필요한 것 같습니다.

최슬기

팀프로젝트를 진행하면서 에러에 대한 두려움을 어느정도 해소한 것 같습니다. 개개인의 개발 능력도 중요하지만 서로 간의 의사소통과 협업 같은 개발 외적인 부분도 매우 중요하다는 것을 깨닫는 의미 깊은 시간이었습니다.

백인준

테이블 설계를 제대로 하지 않아 중간에 컬럼명을 바꾸게 되면서 협업하는 과정에서 치명적인 오류를 유발할 수 있다는 것을 알게 되었습니다. DB 설계를 직관적이고 효율적으로 모델화 하여 완벽하게 설계할 수 있도록 해야 할 것 같습니다.

이슬아

프로젝트를 하면서 최초 기획, 설계와 다르게 변경되는 점이 많아서 기본 설계가 많이 중요하다고 느꼈습니다. 프로젝트를 진행하면서 협업의 중요성을 다시 한번 깨닫게 되었으며, 의사소통을 원활하게 하는 방식을 배울 수 있는 좋은 경험이었습니다.

한채영

전체적인 진행방향, ERP 시스템에 대한 이해, 흐름과 데이터를 DB화 하기 위해 팀원들과의 원활한 의사소통과 회의를 진행함으로써 협업을 통한 커뮤니케이션의 중요성을 알게 되었습니다.



감사합니다.

UP Bank 올림.