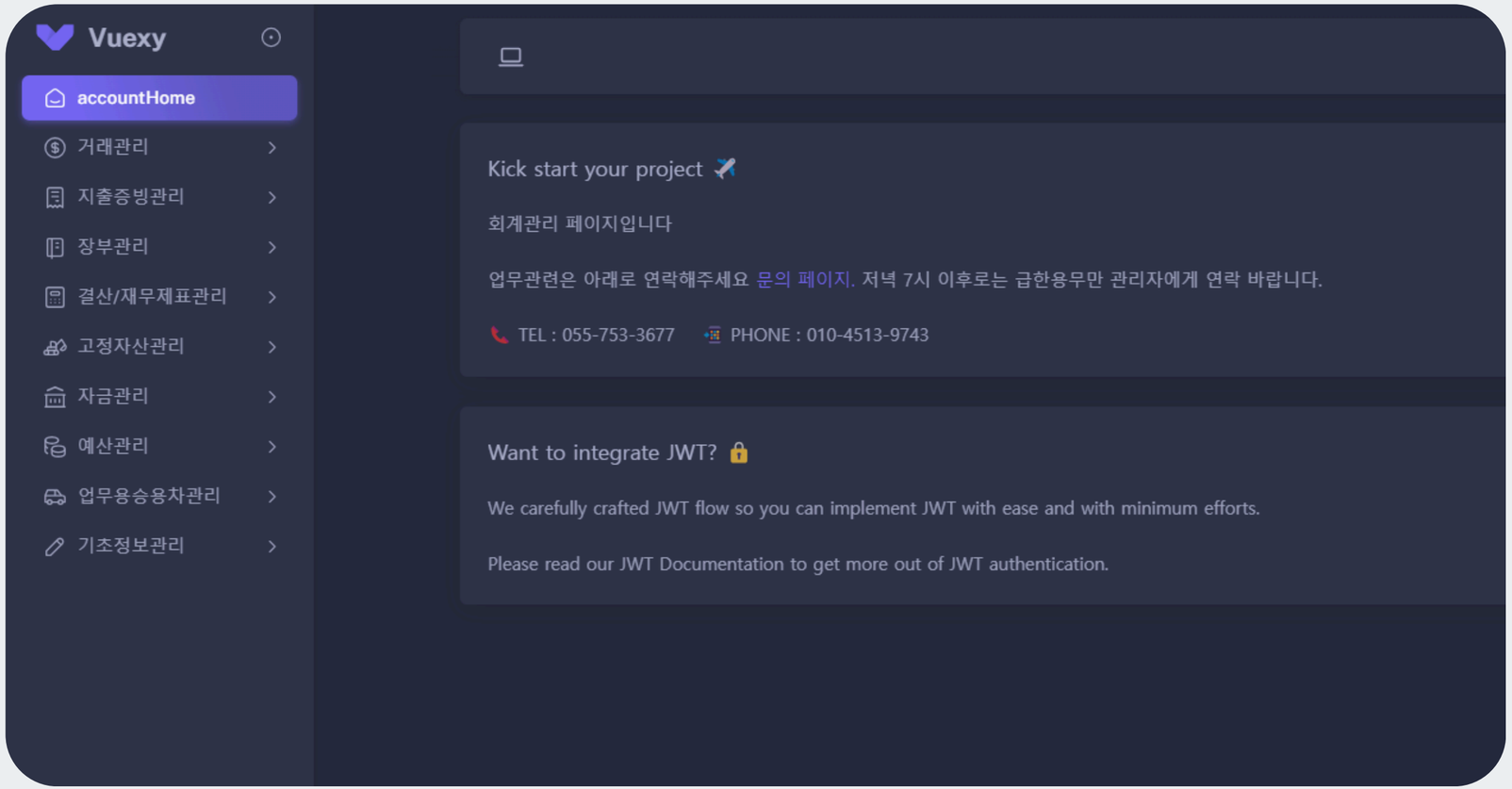
회계 시스템 개발



CONTACT

010-3057-5530

wngus183@naver.com

Git

https://github.com/juhyeonjungo/portfolio-test/blob/main/juhyeonproject.pdf

담당 역할

-거래처별 원장 페이지 및 기능 구현

-기수 관리 페이지 및 기능 구현

-손익 계산서 페이지 및 기능 구현

프로젝트 특징

-Sping Boot를 이용한 백엔드 구현

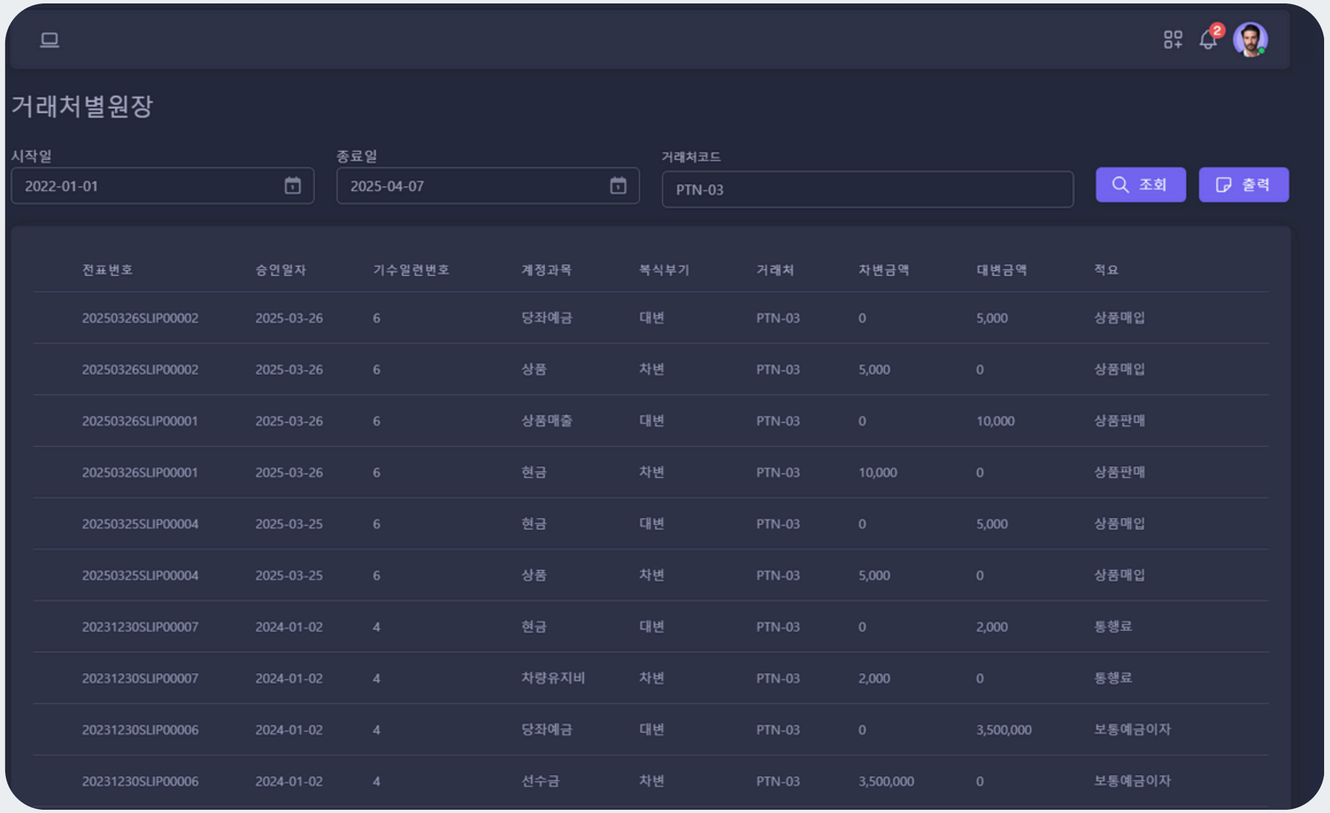
-MyBatis를 활용한 oracle DB연동

-vue 3+Nuxt를 활용한 SPA 기반 프론트엔드 구현

-Axios를 통한 API 연동 및 비동기 데이터 처리

-Sourcetree를 이용한 Bitbucket 협업 및 브랜치 전략 적용

거래처별 원장 페이지



기능 개요

거래처 관리 기능은 회계 시스템에서 필수적인 거래처(고객사) 정보를
효율적으로 관리하기 위한 화면입니다.
사용자는 해당 거래처를 조회 작업을 할 수 있으며,
이를 통해 다양한 회계 문서 작성 시 필요한 거래처 데이터를 제공받게 됩니다.

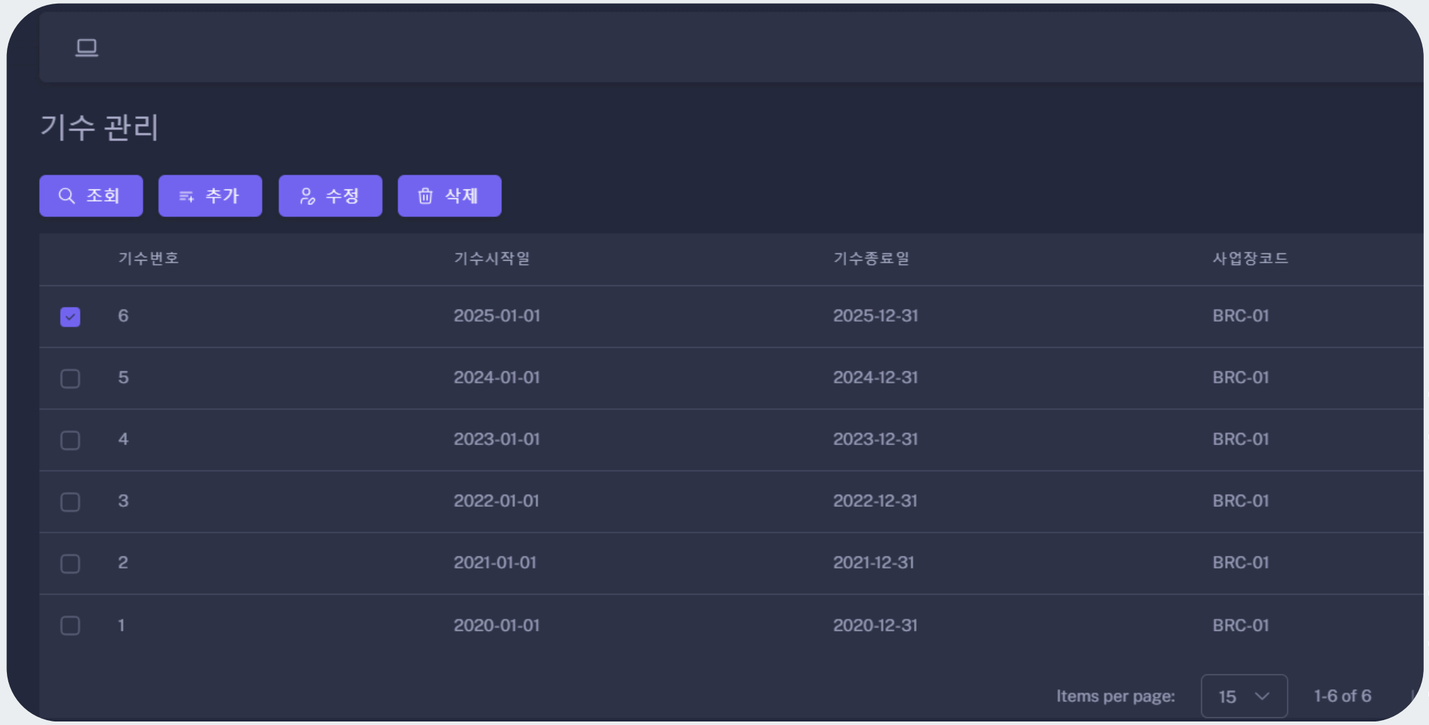
거래처 테이블 섹션

- **VDataTable을 활용해 거래처 목록을 표 형태로 보여줍니다.**
- **거래처코드를 클릭하면 해당 거래처 코드가 선택되어 조회가 가능합니다.**
- **거래처 정보에는 코드,사업장,대표자,사업장등록번호,주소 등 다양한 속성 포함됩니다**

날짜 선택기

- **사용자는 Vuetify의 VDatePicker를 통해 조회 기간을 직접 선택할 수 있습니다.**
- **시작일과 종료일을 설정하면, 해당 기간 동안의 전표 목록만 화면에 표시됩니다.**

기수 관리 페이지



기능 개요

기수 관리는 회계 연도별 시작일과 종료일, 사업장 정보를 등록 및 관리하는 기능입니다.
이 정보는 손익계산서, 재무제표 등 회계 데이터를 분류하는 기준으로 사용됩니다.

조회 기능

- **조회 버튼 클릭시 DB에서 전체 기수 데이터를 불러옵니다.**
- **Vuetify의 DataTable을 활용해 기수 정보를 테이블 형태로 출력합니다.**

추가 기능

- **추가버튼 클릭 시 입력용 모달창이 팝업됩니다.**
- **사용자 입력값: 기수번호,시작일,종료일,사업장코드**
- **확인 버튼 클릭 시 DB에 저장되고,리스트에 실시간 반영됩니다.**

수정 기능

- **테이블에서 수정할 기수를 체크 후 수정 버튼을 클릭**
- **해당 기수의 기존 정보를 불러온 뒤 모달에서 편집 가능**
- **확인 클릭시 DB 업데이트 및 리스트에 반영**

손익계산서 페이지

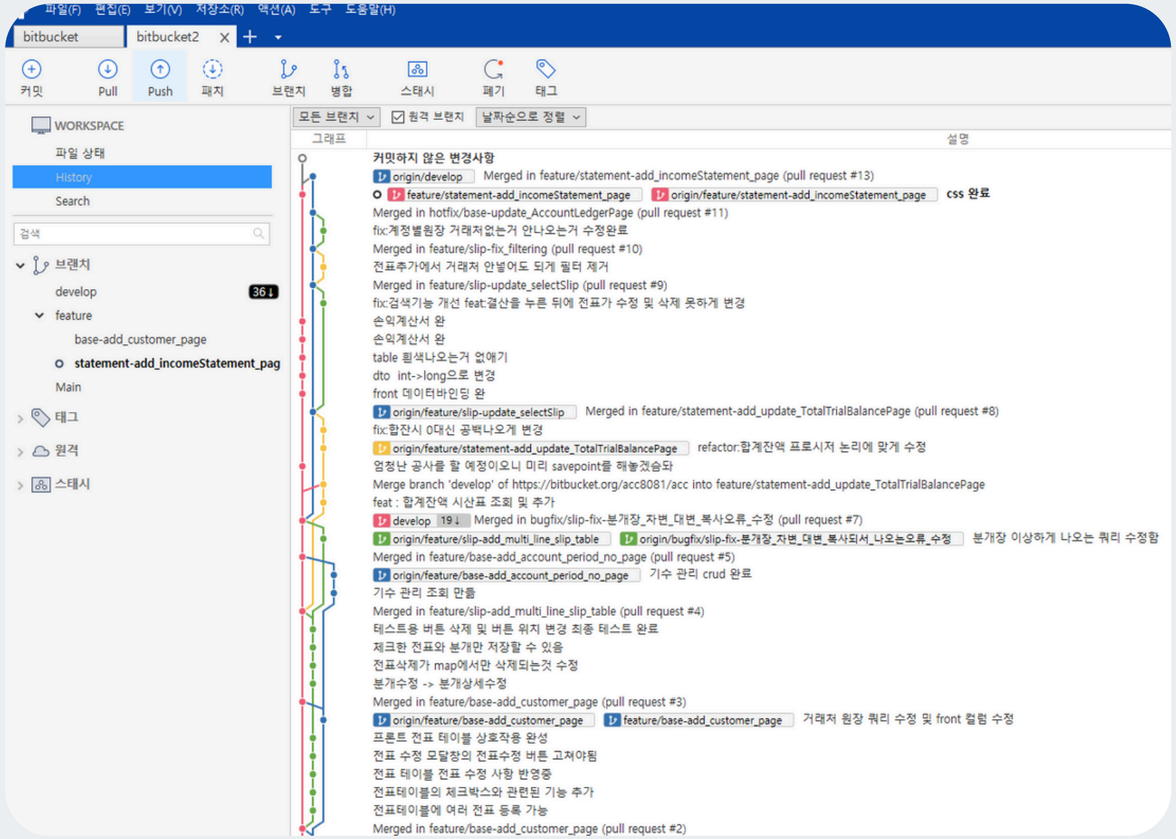
기능 개요

- 손익계산서는 특정 회계연도(기수)를 기준으로 수익과 비용을 비교하여 당기 순이익을 계산하는 화면입니다.
  사용자는 원하는 기수를 선택하여 해당 기간의 손익 데이터를 조회할 수 있습니다.

조회 기능

- 조회 버튼 클릭 시 백엔드로부터 손익 데이터를 가져옵니다.
- 데이터는 Oracle DB에서 계정 과목의 레벨 정보(LEV)를 기준으로 분류되며,
  LEV = 1인 항목(대표 계정 과목)만 프론트에서 굵은 컬럼 스타일로 출력되도록 설정하였습니다.
- 각 항목은 당기 합계금액 / 전기 합계금액 형태로 비교 표시되며,
  Vuetify의 VDataTable을 활용해 표 형태로 UI를 구성하였습니다.

협업 경험



협업 방식

- 3인 팀 프로젝트로 진행되었으며 추가 기능 과정을 함께 참여
- Git + Sourcetree + Bitbucket을 활용한 형상관리 및 협업
- 기능 단위로 브랜치를 분리하고, Pull Request(PR)을 통해 코드 리뷰 및 병합

역할 분담

- 정주현: 거래처별 원장,기수 관리, 손익계산서 페이지 전체 기능 구현
- 오상섭: 분개 페이지 front 기능 구현, 재무상태표 화면 및 기능 개발
- 이길태: 프로젝트 총괄 및 main 브랜치 관리, 합계잔액시산표 구현

브랜치 전략

- develop 브랜치 기반으로 기능별 feature/기능명 브랜치 생성
- 기능 완료 후 pull request로 병합 진행
- Sourcetree에서 커밋 이력과 브랜치 흐름을 시각적으로 관리

협업 중 문제 해결

- Merge 시 충돌이 발생해, 팀원과 함께 테스트용 브랜치 및 레포지토리를 따로 만들어 병합 연습을 진행

느낀 점

- Vue 3 + Nuxt의 조합을 통해 SPA 구조에 적응할 수 있었고, Pinia를 사용하며 상태 관리를 명확히 구분하는 경험을 했습니다.
  특히 모달 창 간 데이터 흐름과 컴포넌트 간 통신을 효율적으로 관리하는 데 도움을 받았습니다.
- Spring Boot + MyBatis + Oracle DB 구조를 직접 다루면서 백엔드에서 요청을 처리하고,
  REST API로 JSON 데이터를 전달받아 UI에 바로 반영하는 흐름을 익혔습니다.
- 기수관리와 거래처 관리 기능에서 데이터의 CRUD 흐름 전체를 직접 구현해보며 MVC 구조의 핵심 로직을 실제로 체득했습니다.
- MyBatis XML을 이용해 쿼리를 직접 작성하며 SQL 문법의 중요성과 쿼리 최적화에 대해 배웠습니다.

주요 에러 및 해결 과정

1. Parameter Binding 에러

- 원인: 프론트에서 보내는 데이터와 백엔드에서 받는 DTO 타입 불일치 (예: list,Map)
- 해결: DTO 클래스와 @RequestBody, @RequestParam 매핑 타입을 일치시켜 해결했습니다.

2. 404 Not Found 에러

- 원인: @RequestMapping, @GetMapping, @PostMapping 등에서 백엔드 URL 경로가 잘못되었거나,
  프론트에서 사용하는 Axios의 요청 주소가 백엔드 주소와 정확히 일치하지 않아서 발생했습니다.
- 해결:
  프론트엔드에서 Axios로 호출하는 주소와 백엔드 컨트롤러의 URL을 정확히 일치시켰습니다.
  백엔드 API 주소(/acc/base/...)가 실제로 존재하는지 확인하고, 그에 맞게 프론트에서 호출 경로를 수정했습니다.

정주현

THANK YOU

CONTACT

010-3057-5530

wngus183@naver.com