SKKUDO: Club Manage Platform

Minseok Yeon, Jaeyoung Kim, Taeju Mun and Seonjong Lee

Sungkyunkwan University - Capstone Design Project

Abstract. 성균관대학교의 모든 학술 동아리 및 학회의 단체 관리 시스템 SKKUDO를 소개하면서 이에 대해 서술한다. Customizable, But Easy to Use를 슬로건으로 내세우는 SKKUDO는 주 고객층으로 동아리 관리자와 동아리원을 타겟으로 설정하였으며, 특히 동아리 관리자에게 필요한 5가지 필수적인 기능인 모집 관리, 일정 관리, 공지 관리, 유저 관리, 권한 관리를 제공한다.

Keywords: 학술 동아리 · 동아리 관리 플랫폼 · Customizable, But Easy to Use.

1 Introduction

동아리 활동은 대학 생활의 꽃이라고도 한다. 학생들은 대학 생활 동안 단순히 동아리 활동에 참여하는 것뿐만이 아니라 동아리를 관리하면서 더 많은 것을 배우고, 소통하고, 자신의 미래에 대해 깊은 고민을 하기도 한다. 하지만 대학교에는 다양한 동아리가 있는 만큼, 주변에 동아리 관리로 어려움을 겪는 학우들을 쉽게 발견할 수 있다. 동아리의 규모가 커지면서 동아리 부원 관리에 어려움을 겪거나, 모집 기간과 같은 동아리가 바쁜 시기 동안 수많은 일정을 조율하는데 불필요한 에너지를 낭비한다.

지금까지의 동아리 관리 플랫폼은 모두 동아리 관리자에게 필요한 서비스를 충분히 제공하지 못했다. 동아리 관리의 내용이 형식화되어 있고, 일부 기능이 빠져 있거나 사용하기 불편하다는 등의 문제점이 있었다. 이러한 문제를 해결하고자 학생들을 도와줄 수 있는 동아리 관리 플랫폼 SKKUDO를 고안하였다.

SKKUDO는 동아리마다 각각의 목적에 맞춰 동아리를 관리할 수 있도록 맞춤형 관리 서비스를 지원한다. 동아리 관리의 필수 기능인 모집 관리, 공지 관리, 일정 관리, 유저 관리, 권한 관리에 더해 가계부 등의 추가 기능까지 모두 포함되어 있다. 특히 우리가 자신 있게 선보이고자 하는 내용인 관리자가 원하는 동아리원의 정보열을 추가하는 혁신적인 기능도 함께 만나볼 수 있으며, 자세한 내용은 Implementation 부분에서 다루도록 한다. 또한, 일반 동아리 부원들은 SKKUDO를 통해 동아리의 공지와 일정을 이전보다 더욱 쉽게 확인하는 것이 가능하며, 아직 동아리에 가입하지 않은 학생들 또한 SKKUDO를 통해 쉽게 학교의 동아리들을 둘러보고 지원할 수 있다. SKKUDO를 통해 학생들은 그동안 겪어왔던 학교 동아리와 관련된 수많은 문제점을 해결할 수 있을 뿐 아니라, 더 나아가 동아리 인원들의 결속력을 강화하여 동아리의 빠른 성장을 도모하는 것 또한 가능할 것으로 예상한다.

SKKUDO는 기획 단계에서 동아리 관리자들을 위한 웹 애플리케이션으로 설계되었으며, 점진적인 구현 과정을 거치면서 위에서 언급한 동아리 관리의 필수적인 기능 외에 일반 동아리 부원을 위한 기능과 가계부 등의 추가적인 기능이 더해지면서 완전한 동아리 관리 플랫폼의 모습을 띠게 되었다. 프론트엔드는 JavaScript 라이브러리의 React, 백엔드는 NodeJS를 통해 개발하였으며, 데이터베이스는 MongoDB를 사용하였다.

2 Design for the Proposed System/Solution/Service

이 장에서는 SKKUDO의 전체적인 시스템과 시스템을 구성하는 서브 시스템의 디자인을 서술한다. 시스템이 어떠한 서브 시스템들로 구성되었는지 각 서브 시스템은 무슨 기능을 하며 서로 간의 상관관계는 무엇인지 살펴보며 SKKUDO의 디자인을 설명한다.

2.1 Overall Architecture

SKKUDO의 시스템은 크게 유저와 상호작용하는 프론트엔드(웹 브라우저) 프론트엔드의 요청을 처리하는 백엔드(서버) 그리고 시스템의 데이터를 저장하는 데이터베이스, 세 파트로 나눌 수 있다. 프론트엔드에서 사용자와의 상호작용에 따른 요청이 서버로 들어오면 서버는 이 요청의 종류에 알맞은 응답을 데이터베이스에서 꺼내와서 프론트엔드에 다시 돌려준다. 전체적인 시스템의 아키텍처는 아래 그림1으로 확인할 수 있다.

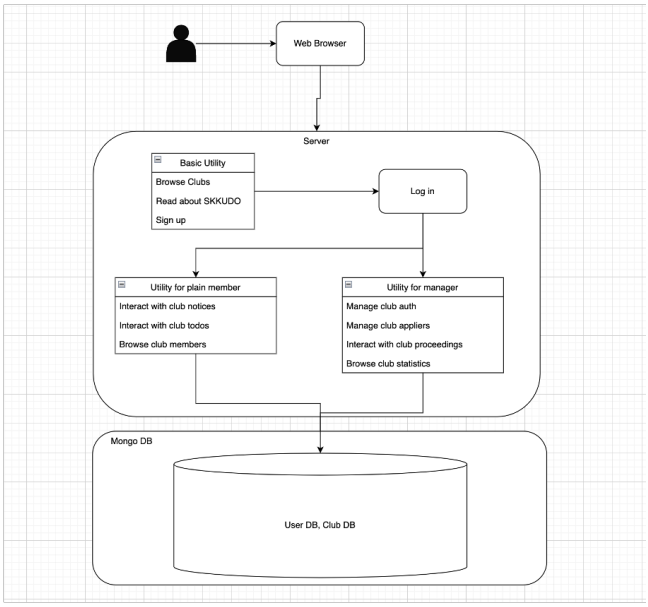

Fig. 1. SKKUD의 전체 시스템 아키텍처

SKKUDO의 서버는 프론트엔드에서 요청을 보내는 사용자에 따라 서브 시스템을 나눌 수 있다. 익명의 사용자를 위한 시스템, 평범한 동아리원을 위한 시스템, 그리고 동아리 관리자를 위한 시스템으로 나뉜다. 익명의 사용자는 동아리 목록을 둘러보고 SKKUDO의 사용법에 대해 읽을 수 있으며 로그인 및 회원가입 시스템을 사용할 수 있다. 사용자가 로그인한 동아리의 평원이라면 해당 동아리의 공지 및 일정을 확인할 수 있으며 같은 동아리원의 명단을 확인할 수가 있다. 마지막으로 사용자가 동아리의 관리자라면 동아리의 권한을 설정할 수 있고 동아리의 모집관리 시스템을 사용할 수 있으며 동아리와 관련된 각종 통계자료를 열람할 수 있고 전반적인 동아리 관리 시스템에 접근할 수 있다. 이렇듯 SKKUDO는 먼저 사용자의 상태에 따라 서브 시스템을 구분하면서 더욱 안전하고 명확한 시스템 아키텍처를 설계했다.

2.2 Front-end Architecture

유저와 서버 사이에서 요청을 받고 응답을 회신하는 프론트엔드의 전반적인 아키텍처를 소개한다. SKKUDO의 프론트엔드는 아래 그림처럼2 크게 로그인하지 않은 사용자와 로그인 한 사용자가 접하는 서브 시스템 두 파트로 나눌 수가 있다. 로그인하지 않은 사용자가 시스템을 사용하려고 할 때 SKKUDO는 로그인의 여부만 판단하고 만약 사용자가 로그인했다면 로그인의 여부와 더불어 해당 동아리가 가지고 있는 권한 테이블을 바탕으로 모든 활동을 인증 및 인가한다.

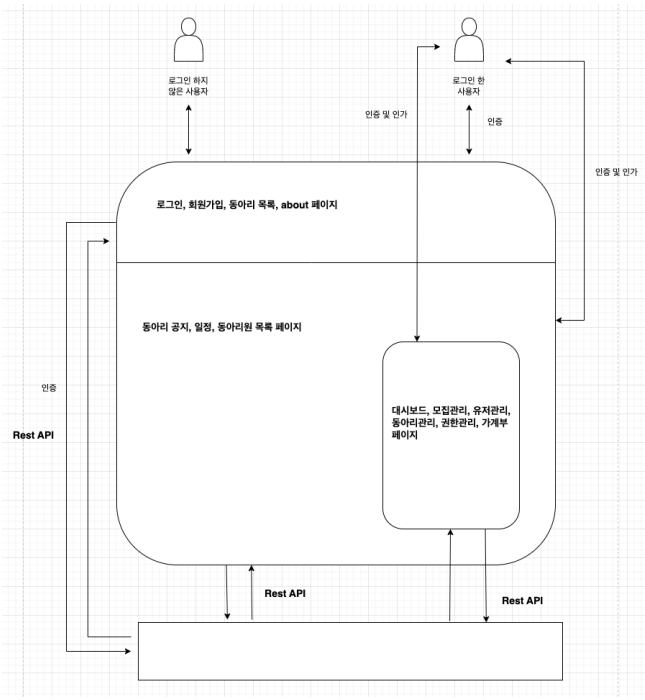

Fig. 2. Front-end architecture

유저의 로그인 여부는 서버와의 쿠키 전송을 통해 판별하여 만약 로그인하지 않은 사용자가 로그인이 필요한 페이지에 접근하려 한다면 로그인 페이지로 리다이렉트 한다. 유저의 권한 인증은 서버의 별도 인증 함수를 통한 검증을 통해 해당 유저가 권한에 맞는 행동을 하는지 판별하며 만약 사용자가 권한을 넘어서는 페이지에 접근하려 하거나 행동을 취한다면 에러 메시지를 띄우며 사용자의 행동을 막거나 취소한다.

다음으로는 SKKUDO의 기능에 따라 나눈 서브시스템을 나열하고 각 서브시스템의 기능과 디자인에 대해 서술하겠다.

2.3 공지관리

SKKUDO에서는 공지의 생성, 삭제, 수정, 읽기를 지원한다. 또한 공지마다 포함하는 카테고리도 추가하여 공지의 필터링과 명확한 구분이 가능하다. 공지는 제목, 작성자, 카테고리, 공개/비공개 여부, 본문으로 구성되어있다. 제목은 공지의 전체 내용을 포괄하고 요약하는 짧은 문장이어야 하며 작성자는 해당 공지를 작성한 사용자의 이름, 카테고리는 해당 공지가 포함할 사전에 정의된 카테고리, 공개/비공개 여부는 해당 공지가 권한에 따라 읽기가 제한될 것인지, 그리고 마지막으로 본문은 해당 공지의 내용이다. 공지는 권한에 따라 읽기, 삭제, 수정, 생성이 제한된다. 비공개인 공지를 읽을 수 있는 권한은 공지 쓰기 권한을 따라가며 공개인 공지는 공지 읽기 권한을 따라간다.

2.4 일정관리

SKKUDO에서는 일정의 생성, 삭제, 수정, 읽기를 지원한다. 또한 일정마다 포함하는 카테고리도 추가하여 공지의 필터링과 명확한 구분이 가능하다. 일정은 제목, 내용, 공개여부, 시작 시간, 종료 시간, 참여 동아리원, 일정 카테고리로 이루어져 있다. 제목은 일정의 내용을 요약하는 짧은 문장이며 내용은 해당 일정의 세부 내용, 공개/비공개 여부는 해당 일정이 권한에 따라 읽기가 제한될 것인지 시작 시간은 해당 일정이 시작하는 시간, 종료 시간은 해당 일정이 종료되는 시간, 참여 동아리원은 해당 일정의 참여하는 동아리원들의 명단, 그리고 마지막으로 카테고리는 해당 일정이 포함할 카테고리의 목록이다. 일정은 권한에 따라 읽기, 삭제, 수정, 생성이 제한된다. 비공개인 일정을 읽을 수 있는 권한은 일정쓰기 권한을 따라가며 공개인 일정은 일정 읽기 권한을 따라간다.

2.5 유저관리

SKKUDO에서는 관리하는 동아리에 속한 동아리원들의 정보를 관리할 수 있다. 먼저 동아리원이 처음 동아리에 지원했을 때 제출한 지원서의 내용과 회원 가입했을 때의 정보를 바탕으로 이름, 학번, 역할, 학과, 소속 학교 위치, 연락처를 열람할 수 있으며 이 정보들은 관리자가 임의로 수정하거나 삭제할 수 없다. 앞서 나열한 정보들 외에 관리자가 동아리원들을 통해 추가로 받고 싶은 정보가 있다면 추가 정보 항목을 생성하거나 수정하고 삭제할 수 있다. 추가된 정보는 유저 관리의 다른 정보들과 함께 보여지며 문자열, 숫자, O/X의 형식으로 동아리원의 정보를 설정할 수 있다. 해당 추가 정보는 다음에 서술할 모집관리 부분의 모집서에 자동으로 추가되어 추가질문으로서의 역할을 한다. 또한 해당 동아리원의 역할을 수정하거나 동아리에서 방출하는 것 또한 유저 관리 시스템에서 가능하다.

2.6 모집관리

SKKUDO에서는 동아리에 지원한 지원자들과 동아리의 모집 기간 동안 사용되는 지원서 양식을 관리할 수 있다. 지원서 양식은 크게 서류질문, 추가 질문, 면접 질문으로 나뉘며 추가 질문은 앞에서 서술한 유저의 추가정보가 자동으로 넘어와서 생성된다. 지원서는 생성, 수정, 삭제, 읽기가 가능하며 만약 해당 지원서로 지원을 한 지원자가 있다면 지원서를 수정하거나 삭제하는 것은 불가능하다. 지원자들의 명단에서는 회원가입 시 저장되었던 지원자의 이름, 학번, 학과, 연락처가 자동으로 등록되며 지원서에서 질문한 서류 질문들과 면접 질문들, 그리고 추가 질문들에 대한 답변을 확인할 수 있다. 추가 질문에 대한 답변을 제외한 모든 답변에 점수를 매길 수 있으며 점수를 기재하면 서류 점수들의 총합, 면접점수들의 총합, 그리고 전체 답변 점수의 총합이 자동으로 계산된다. 또한 SKKUDO의 모집관리에서는 관리자가 설정한 서류 점수, 면접 점수의 반영 비율과 합격시킬 인원수에 기반하여 상위권 점수를 보유한 지원자들을 나열하여 일괄적으로 합격시킬 수 있는 자동 합격기능도 존재한다.

2.7 동아리관리

SKKUDO에서는 관리자가 동아리의 정보에 접근하고 수정할 수 있다. 접근 및 수정이 가능한 동아리의 정보로는 동아리 이름, 동아리 주제, 동아리가 속한 학교의 위치, 동아리의 모집 형태, 동아리의 모집 기간 시작일, 동아리의 모집 기간 종료일이다. 만약 동아리가 상시 모집이라면 동아리의 모집 기간을 설정할 필요가 없다. 또한 동아리 목록 페이지에서 동아리의 이름과 주제를 비롯해 함께 보일 동아리의 대표 이미지 또한 업로드하여 설정이 가능하다. 동아리가 정규모집인 동시에 사용자가 동아리 목록 페이지에 접속한 날짜가 사전에 설정한 모집 기간 안에 속해 있다면 해당 동아리는 동아리 목록 페이지에서 “현재 모집 중인 동아리”로 분류되어서 자연스러운 홍보 효과를 가져갈 수 있다.

2.8 가계부관리

SKKUDO에서는 동아리의 가계부를 관리할 수 있다. 한 동아리당 하나의 가계부를 최대로 생성할 수 있으며 삭제, 수정, 읽기가 가능하다. 가계부는 동아리와 관련된 금전 행위를 표의 형태로 표시한 것이며 행에는 해당 행위를 세분화하는 항목들과 열에는 해당 행위가 나타나 있다. 행에 있는 항목으로는 날짜, 수입, 지출, 누가, 내용, 잔액이 있으며 추가적인 내용으로는 계좌번호와 메모가 존재한다. 가계부에서 새로 열을 하나 생성하려면 우선 기본값으로 세팅된 빈 열을 하나 추가한 다음 정보를 수정하면 되는 형태이다. 열 마다 삭제가 가능하며 가계부 전체를 한 번에 삭제하는 것 또한 가능하다.

2.9 권한관리

SKKUDO에서는 다양한 권한을 통해 동아리원의 행동에 안전장치를 걸어두고 이를 통한 세심한 동아리 관리가 가능하게 한다. 관리자가 설정할 수 있는 권한의 목록은 다음과 같다. 공지 읽기 권한, 공지 쓰기 권한 & 숨김 공지 열람 권한, 동아리원 열람 권한, 동아리원 수정 권한, 동아리원 정보 항목 추가 권한, 일정 읽기 권한, 일정 쓰기 권한 & 숨김 일정 열람 권한, 모집 서류 읽기 권한, 모집 서류 수정

권한, 권한 읽기 권한, 권한 수정 권한, 동아리 정보 읽기 권한, 동아리 정보 수정 권한, 가계부 열람 권한, 가계부 수정 권한, 모집 지원자 열람 권한, 모집 지원자 열람 권한, 모집 지원자 수정 권한, 카테고리 수정 권한 총 18개의 권한을 관리할 수 있다. 각 항목에 따른 권한은 회장까지, 부회장까지, 운영진까지, 모든 사람이 총 4단계로 나뉜다. 유저가 이 권한 관리시스템에서 설정한 권한에 반하는 행동을 시도할 경우 해당 행위는 취소되며 에러 메시지가 나타난다.

2.10 대시보드

SKKUDO의 대시보드 시스템은 해당 동아리의 정보를 종합하여 이를 통계자료와 그래프로 시각화하여 사용자에게 한 화면으로 보여주는 시스템이다. 동아리 부원의 수, 지원자의 수, 금일 동아리의 남은 일정의 개수, 이번 달 동아리의 남은 일정의 개수를 우선 시각자료로 보여주어 사용자로 하여금 동아리의 현재 상황에 대해 더욱 빠르게 파악할 수 있도록 도와준다. 또한 동아리 부원들의 학번 분포와 학과 분포를 그래프를 통해 시각화하여 관리자로 하여금 동아리 부원들의 대한 이해도를 높이고 더욱 효율적인 동아리 관리가 가능해질수 있도록한다. 또한 추가로 유저관리 서브 시스템에서 생성한 해당 동아리의 추가 유저 정보에 관한 그래프도 그려주어 더욱 더 자세한 동아리 관리가 가능하다.

3 Implementation

이 장에서는 앞에서 설명한 시스템의 디자인을 구현하기 위해 수행한 실제 개발 방법을 설명한다. 시스템의 개발은 크게 프론트엔드 파트와 백엔드 파트로 나눌 수 있으며 이 장의 내용 또한 두 파트로 나누어 서술한다.

3.1 Front-end Implementation

Overall UI/UX

SKKUDO를 웹으로 구현할 때 사용한 UI/UX의 컨셉은 성균관대학교이다. 이름에서부터 알 수 있듯이 SKKUDO는 성균관대학교 동아리를 타겟으로 한 시스템이기 때문에 사용자인 성균관대 학생들에게 익숙한 디자인을 통해 친밀감을 형성하고 이를 바탕으로 시스템을 처음 사용할 때 생기는 진입장벽을 낮추는 것을 목표로 한다.

이에 따라서 SKKUDO에 사용된 메인 색상또한 성균관대와 관련이 있는 색상으로 선정하였다. 2022년 성균관대학교 가을축제인 ESKARA의 테마색상이었던 진녹색과 성균관대학교 휘장에 나타나있는 황금색 색상을 메인 색상으로 선정하였으며 진녹색의 HEX코드는 #0c4426, 그리고 황금색의 HEX코드는 #dde143이다. 실제 색은 아래 그림을3, 4을 통해 확인할 수 있다.

전체적인 컴포넌트의 디자인은 메테리얼 디자인으로 구현하여 사용자에게 입체감을 선사하고 더욱 만족스러운 사용자 경험을 제공한다. 더 완성되고 복잡한 메테리얼 디자인을 웹에 그릴 수 있도록 개발 과정에서 MUI 라이브러리를 사용했으며 이를 통해 메테리얼 디자인팀이 제시하는 여러 가지 컴포넌트를 상대적으로 쉽게 구현한다.

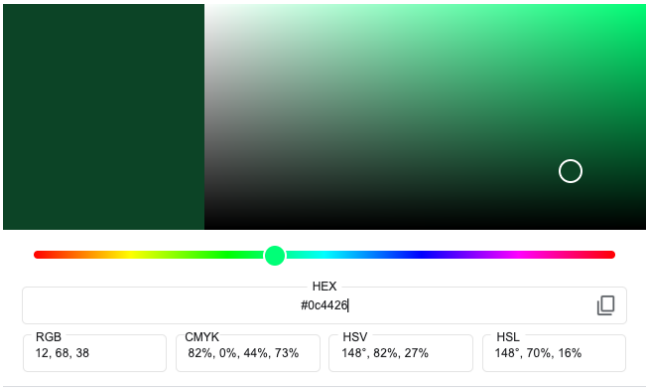

Fig. 3. SKKUD의 메인 색상으로 사용된 진녹색

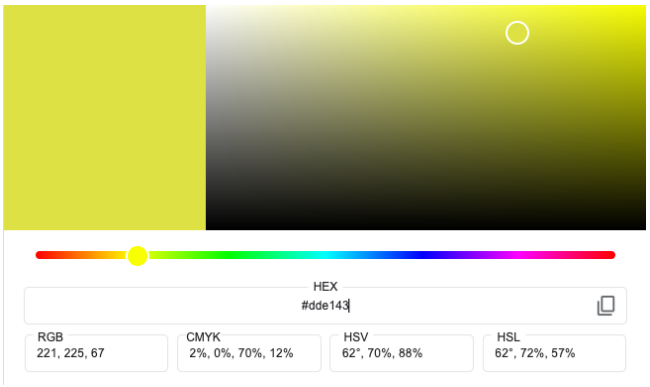

Fig. 4. SKKUD의 메인 색상으로 사용된 황금색

SKKUDO가 추구하는 전문적인 동아리 관리 플랫폼의 이미지를 구축함과 동시에 더 나은 사용자 경험을 위해 관리자가 사용하는 기능들은 관리자 페이지로 따로 구분하여 사용자에게 보여진다. 관리자 페이지는 다른 페이지와는 달리 화면 상단에 위치한 내비게이션 바를 없애고 대신 화면 왼쪽에 사이드바를 추가하여 사용자가 동아리 관리기능에 더 몰입할 수 있도록 하였다.

Libraries Used

React [1] SKKUDO 프론트엔드에서는 메타에서 개발한 자바스크립트 웹 라이브러리인 리액트를 사용한다. 리액트를 통해 더 손쉽고 빠른 개발 속도를 확보하였으며 리액트의 특징인 virtual DOM을 활용한 페이지 업데이트를 통해 시스템의 높은 퍼포먼스를 이루어냈다. 또한 후술할 리액트와 관련된 다른 여러 가지 라이브러리들과 함께 사용함으로써 긍정적인 시너지 효과를 내었으며 모듈형 개발을 통해 높은 생산성 또한 이루어낼 수 있었다.

Styled Components [2] 리액트가 사용하는 컴포넌트 시스템의 스타일링을 더욱 효율적으로 수행하기 위해 styled component 라이브러리를 사용한다. 해당 라이브러리를 통해 CSS 코드를 자바스크립트 파일 안에서 작성함으로써 클래스 이름 버그를 최소화하고 컴포넌트와 해당 스타일 간의 연결고리를 최소화하여 의존성을 낮춤과 동시에 재생산성을 높였으며 자바스크립트 코드의 동작에 따른 스타일 변화를 더욱더 쉽게 구현한다.

Axios & React Query [3][4] 백엔드와의 효율적인 REST API 통신을 위하여 Axios와 React Query 라이브러리를 사용한다. Axios는 node.js와 브라우저를 위한 HTTP 클라이언트이다. 요청 및 응답을 자동으로 반환해주는 것뿐만이 아니라 반환받은 JSON 형태의 데이터를 자동으로 타입 변환을 해주어 편리한 개발이 가능하다. React Query는 서버와의 API 통신을 위한 다양한 기능들을 제공한다. API 호출의 성공과 실패 각각의 상황에서의 효율적인 대응을 가능하게 하며 호출 함수의 연계 또한 가능하게 하여 매끄러운 코드 연결을 작성한다. 이러한 React Query 라이브러리와 Axios 라이브러리를 동시에 사용함으로써 프론트엔드 파트는 생산성과 효율성을 극대화하였다.

React Router Dom [5] SKKUDO에는 20개가 넘는 페이지가 존재한다. 이 페이지들을 연결하고 각 페이지에 알맞은 URL의 형태를 제공하기 위해서 React Router Dom 라이브러리를 사용한다. 또한 비슷한 형태의 URL을 공유하는 페이지들끼리는 Nested Route를 통해 URL과 컴포넌트 계층구조를 연결하면서 더욱 효율적인 페이지의 구성을 연출한다.

React Icons [6] React Icons는 리액트에서 사용할 수 있는 여러가지 아이콘들을 제공하는 라이브러리이다. 해당 라이브러리를 통해 SKKUDO에서 필요한 아이콘을 불러와서 사용함으로써 사용자에게 더욱 친숙한 인터페이스와 자연스러운 UI를 연출하였다.

React Calendar [7]

SKKUDO의 일정 관리 부분에서는 날짜에 따라서 일정을 분류하고 관리한다. 이를 사용자 입장에서 더 직관적으로 파악할 수 있도록 React Calendar 라이브러리를 통해 완성된 디자인의 달력을 화면에 구현한다. 라이브러리를 통해 불러온 달력 컴포넌트에 SKKUDO의 테마 색상들을 섞어 적절한 SKKUDO만의 달력을 제공함으로써 일정 관리의 편의성과 유용성을 증대시킨다.

Apex Chart [8]

SKKUDO의 대시보드 페이지는 사용자가 해당 동아리에서 수집한 데이터를 바탕으로 그려진 각종 통계자료와 그래프를 확인할 수 있다. 이때 그래프를 그릴 때 사용한 라이브러리가 Apex Chart이다. SKKUDO는 Apex Chart 라이브러리에 존재하는 Bar, Line, Pie chart를 사용하였으며 이를 통해 데이터를 시각화하고 표현하는 데 도움을 얻었다.

MUI [9]

MUI 는 메테리얼 디자인의 완성된 컴포넌트를 불러올 수 있는 라이브러리이다. SKKUDO의 전체적인 메테리얼 디자인 컨셉에 맞게 여러 가지 컴포넌트를 MUI에서 불러와서 시스템에 적용한다. 예를 들어, 카드 모양의 하얀색 배경을 가진

컴포넌트를 구현하고 싶다면 MUI 의 컴포넌트를 불러와서 사용하면 된다. 이렇듯 이미 완성된 디자인의 컴포넌트를 사용함으로써 빠른 개발 속도와 디자인 측면에서 높은 완성도를 챙길 수 있다.

Framer Motion [10]
SKKUDO의 더욱 만족스러운 사용자경험을 위해 애니메이션 라이브러리인 Framer Motion 라이브러리를 사용한다. 해당 라이브러리를 통해 한층 더 고급스러운 애니메이션 효과를 컴포넌트에 부여할 수 있으며 조금 더 매끄럽고 섬세한 움직임을 구현할 수 있다. 예를 들어, 사용자가 버튼에 마우스를 올려놨을 때 버튼의 크기를 바꾸거나 특정 컴포넌트에 마우스를 올려놓았을 때 해당 컴포넌트의 배경 색상이 바뀌는 등 사용자의 현재 동작에 대해 애니메이션으로 간접적으로 알려줌으로써 높은 수준의 편의성과 심미성을 제공한다

Recoil [11]
Recoil은 React를 위한 상태관리 라이브러리이다. React에서는 recoil 말고도 redux와 같은 비슷한 기능을 하는 다른 상태관리 라이브러리들이 존재하지만 그 중 Recoil을 선택한 이유는 해당 라이브러리의 간편함과 사용함에 있어서 편의성이 높기 때문이다. 상대적으로 사용 방법이 쉽고 진입장벽이 낮은 Recoil 라이브러리를 개발단계에 사용함으로써 같은 팀원들의 낮은 러닝 커브를 유지하고 동시에 간편하게 상태관리를 해내면서 코드의 높은 유지보수성을 이뤄냈다. SKKUDO에서는 로그인한 유저의 정보나 현재 들어와 있는 동아리 페이지의 동아리 정보 등을 Recoil을 통해 관리한다.

페이지 구성

SKKUDO는 총 28개의 서로 다른 URL과 매칭되는 페이지로 구성된다. React Router Dom을 통해 서로 연결되며 Nested Route로 연결된 3개의 페이지 덩어리가 존재한다. 이러한 특징을 가진 SKKUDO의 페이지들은 크게 세 파트로 나눌 수 있다. 익명의 사용자를 위한 페이지, 평범한 동아리원을 위한 페이지, 그리고 동아리 관리자를 위한 페이지로 나뉜다. 이는 사용자가 현재 사용하고 있는 SKKUDO의 서비스에 따라 디자인을 달리하여 명시적인 구분 점을 주기 위함이다. 또한 파트에 따라 사용자 권한 인증과 같은 필요한 함수 호출이 다르기 때문에 이러한 구분을 두었다. 평범한 동아리원이란 권한이 제한된 동아리원을 의미하며 이때 권한은 동아리마다 상이하다. 따라서 현재는 서술의 용이함을 위해 권한에 구애받지 않는 제일 낮은 권한인 "부원" 사용자가 공통으로 사용할 수 있는 페이지를 평범한 동아리원을 위한 페이지라고 정의한다. 지금부터 각 파트에 대한 설명을 서술한다.

Pages for anonymous users 익명의 사용자란 아직 로그인하지 않은 사용자를 의미한다. 익명의 사용자는 SKKUDO의 홈페이지와 SKKUDO에 등록되어 있는 동아리의 목록을 볼 수 있는 페이지, 회원가입 페이지, 로그인 페이지, 그리고 SKKUDO를 소개하는 About페이지에 접속할 수 있다.

Pages for plain members 일반 동아리원들이 접근할 수 있는 페이지로는 앞에서 설명한 익명의 사용자를 위한 페이지를 포함한 마이페이지, 동아리 공지페이지, 동아리 일정 페이지, 동아리 멤버 페이지가 있다. 마이페이지에서는 접속한 계정의

사용자 정보와 사용자가 속한 동아리들의 목록, 그리고 사용자가 지원하여 결과를 기다리고 있는 동아리들의 목록을 확인할 수 있다. 마이페이지에서 자신이 속한 동아리를 클릭하면 바로 동아리 공지페이지로 이동하게 된다. 동아리 공지 페이지에서는 지금까지 작성된 동아리의 공지와 공지에 따른 카테고리를 확인할 수 있으며 공지의 세부 내용을 읽을 수 있다. 또한 해당 페이지에서 공지를 수정하거나 생성하는 페이지로 이동할 수 있으며 만약 동아리 공지 쓰기 및 수정 권한이 사용자의 권한보다 높게 설정되어 있다면 해당 페이지로는 이동이 되지만 페이지 내에서 액션이 제한된다. 동아리 일정 페이지에서는 공지페이지와 비슷하게 지금까지 생성된 일정들을 달력 형태로 확인할 수 있으며 일정마다 포함하고 있는 카테고리를 확인할 수 있다. 해당 페이지에서 일정을 삭제, 수정, 및 생성을 할 수 있게 되며 마찬가지로 앞서 말한 행위에 권한이 없다면 액션이 제한된다. 동아리 멤버페이지에서는 해당 동아리의 동아리원들 목록을 확인할 수 있는 페이지이다.

Pages for club managers 관리자를 위한 페이지는 앞서 서술한 일반 동아리원들을 위한 페이지에서 내비게이터에 있는 "동아리 관리" 내비게이션을 타고 들어올 수 있다. 해당 파트는 앞서 말한 두 파트와 달리 화면 상단에 위치한 상단 바가 존재하지 않고 대신 사이드바를 비치하여 전반적으로 다른 디자인을 연출하였다. 사이드바에는 다양한 페이지로 이동할 수 있는 내비게이터가 있으며 이동할 수 있는 페이지로는 대시보드 페이지, 유저 관리 페이지, 모집관리 페이지, 권한관리 페이지, 동아리 관리 페이지, 가계부 페이지가 있다.
대시보드 페이지에서는 동아리의 데이터를 가공하여 그린 여러 가지 통계 자료와 그래프가 사용자에게 보여진다. 기본적으로 4개의 통계 시각자료(그림5)와 2개의 그래프(학번분포 그래프, 학과분포 그래프)(그림6)가 존재하며 추가적으로 동아리마다 가지고 있는 유저의 추가정보에 따른 그래프가 그려지게 된다.
유저관리 페이지에서는 동아리원들의 역할을 수정하고 방출하거나 동아리가 원하는 동아리원들의 추가정보를 생성하는 페이지이다. 이 때 이 추가정보는 앞서 말한 추가 그래프를 그릴 때 사용되거나 뒤에 서술할 모집관리 페이지에서 지원서의 추가 질문을 자동으로 생성할 때 사용된다.
모집관리 페이지에서는 해당 동아리에 지원한 지원자들의 정보를 확인하고 지원서의 답변에 따른 점수를 기입하고 지원자들을 최종적으로 합격 및 불합격시킬 수 있다. 또한 해당 동아리가 동아리 모집 시 사용하는 지원서의 양식을 수정할 수 있으며 삭제하거나 생성할 수 있다. 지원서에 자동으로 추가되는 추가 질문은 앞서 유저관리 페이지에서 관리자가 추가한 해당 동아리가 원하는 동아리원들의 추가정보를 자동으로 반영한 것이다.
권한관리 페이지에서는 동아리원들의 각종 행위를 권한에 따라 제한하는 페이지이다. 총 18개의 항목이 존재하며 권한은 회장까지, 부회장까지, 운영진까지, 모든 사람이, 총 4개의 권한중 하나로 설정할 수 있다.
동아리 관리 페이지에서는 동아리의 정보를 수정할 수 있으며 해당 동아리의 모집기간을 설정하여 모집 중인 동아리에 동아리 이름을 올릴 수가 있다. 또한 동아리의 목록을 볼 수 있는 페이지에서 자신의 동아리를 대표하는 이미지를 설정할 수 있다.
마지막으로 가계부 페이지에서는 해당 동아리의 가계부를 생성하고 삭제하고 수정할 수 있다. 가계부에서는 입금, 지출, 내용, 계좌번호, 잔액, 메모 등의 항목들을 지원한다.

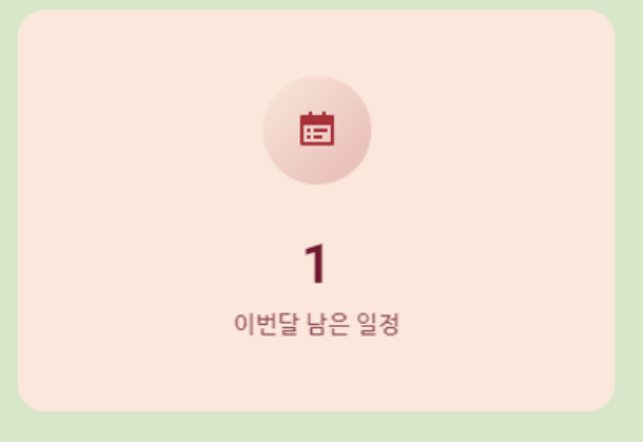

Fig. 5. 대시보드에 보여지는 통계 시각 자료 예시

Fig. 6. 대시보드에 보여지는 학과분포 그래프

3.2 Back-end Implementation

Overall Architecture SKKUDO는 동아리 관리에 필수적인 모집관리, 유저관리, 일정관리, 공지 관리, 권한 관리 등 다양한 기능을 담고있다. 이중 핵심이라고 볼 수 있는 유저의 열(feature)를 동아리마다 마음대로 커스터마이징 할 수 있는 기능을 어떻게 구현하는지가 관건이었다. 해당 기능을 구현하기 위해 자유도가 높은 비관계형데이터베이스, MongoDB를 채택했다.

동아리 관리 플랫폼의 특성상 중요한 개인 정보가 오고 가기 때문에 높은 보안성 또한 프로젝트의 핵심 과제였다. 비밀번호 암호화를 위해 현재 가장 강력한 bcrypt 암호화 알고리즘을 사용했으며, 프론트와 백엔드의 통신 도중에 데이터가 유출되지 않도록 AWS EC2의 로드밸런서 서비스를 통해 https 설정을 해주었다.

관리 플랫폼의 특성상 다양한 상황에서 유저를 인증하는 알고리즘이 필요했다. 그 예시로는 유저의 동아리 내 권한에 따라 동아리 관리의 기능을 제한하는 Validation Table, 동아리원을 수정 및 삭제하는 행위를 수행하는 사용자의 역할과 대상이 되는 사용자의 역할을 비교하여 동아리 관리의 기능을 제한하는 UserRole Hierarchy, 동아리의 지원서와 동아리의 정보 사이의 관계에서 데이터 오염을 방지하는 UserColumn Hierarchy 등이 있다. 해당 내용은 이후 Architecture Detail에서 자세하게 서술한다.

Libraries Used

NodeJs [12] NodeJs는 엄밀히 말하자면 라이브러리는 아니나 웹 밖에서 javascript 코드가 작동할 수 있도록 도와준다. 이를 통해 웹 외부에 존재 하는 백엔드에서 javascript의 코드를 직동할 수 있었다.

Express [13] Express 라이브러리를 통해 API를 제작하고 라우팅하는 시간을 훨씬 단축할 수 있었다. 순수한 NodeJs보다 훨씬 적은 코드로 같은 결과물을 만들었다.

nodemon [14] 변경 사항을 저장하면 자동으로 서버를 새로고침해주는 라이브러리이다. 변경 사항 적용을 위해 수동으로 서버를 껐다가 켜야 하는 수고로움을 덜어준다.

mongoose [15] SKKUDO는 데이터베이스로 MongoDB를 사용한다. 하지만 프로젝트 개발 도중 직접 MongoDB 명령어로 데이터를 처리한다면 코드의 가독성이 좋지 않고, 유지 보수하기가 어려워진다. Mongoose 라이브러리는 MongoDB 명령어로 직접 코딩하는 대신 Javascript(typescript)로 모델을 구성하도록 도움을 준다. Mongoose middleware, mongoose methods 등 다양한 편의 기능을 제공해 더 빠른 서비스 개발이 가능하다.

CORS [16] 웹의 정책상 도메인 이름이나 포트 번호가 다르면 데이터를 서로 주고받을 수 없다. 즉, 포트 번호가 다른 프론트엔드와 백엔드 간의 정보 교환은 불가능하다. 이를 해결하기 위해 나온 정책이 CORS(cross-origin resource sharing)이다. 정보 교환을 허용할 특정 URL을 명시해두면 해당 URL과의 정보 교환을 특별히 허용해준다. Nodejs의 cors 라이브러리를 활용해서 백엔드가 정보를 교환할 수 있는 프론트엔드의 주소를 명시하여 개발 과정에서 도메인이 다른 백엔드와 프론트엔드 간의 데이터 통신을 가능하게 했다.

cookie-parser [17] 프론트엔드와 백엔드가 교환하는 쿠키는 유저의 로그인 여부와 같은 중요한 인가 정보를 담고 있다. 하지만 nodeJs는 이를 해석하지 못하므로 cookie-parser 미들웨어를 설정해 nodeJs가 쿠키를 이해하도록 번역한다.

dotenv [18] 만약 로그인 ID와 같은 중요한 정보를 그대로 GitHub 등에 올리게 된다면 해킹의 위험에 빠질 수 있을 것이다. 그렇기에 이를 비밀리에 환경변수라는 공간에 관리하는데, 환경변수에 있는 데이터를 읽을 수 있도록 도와주는 라이브러리가 dotenv이다.

bcrypt [19] 유저의 비밀번호를 암호화하기 위한 최신 암호화 알고리즘이다.

multer [20] 백엔드에 이미지를 저장하기 위한 라이브러리다. 저장되는 이미지들은 루트 폴더 아래의 uploads라는 이름의 폴더에 들어가게 된다.

Architecture Detail

SKKUDO의 다양한 동아리 관리 기능을 지원하기 위해 다양한 알고리즘 과 데이터구조가 쓰였다. 아래는 개발부터 배포까지의 그 상세한 예시를 다룬다.

User Data Structure 데이터 구조를 설계할때 가장 신경을 썼던 부분은 유저의 열을 편집하는 부분이었다. 동아리마다 부원들에게 추가적으로 받는 열(정보)이 있고, 동아리원은 그 추가되는 열에 동아리마다 답변을 해야했다.

```
export interface Club {
  _id: Types.ObjectId;
  name: string;
  initializer: User;
  location: Location;
  accepted: boolean;
  type: ClubType;
  userColumns: Column[];
  //description
  image: string; //image가 저장된 경로
  recruitType: RecruitType;
  recruitStart: Date | null; //모집 시작일
  recruitEnd: Date | null; //모집 종료일
  createdAt: Date;
  updatedAt: Date;
}
```

Fig. 7. 동아리의 데이터구조도. userColumns로 동아리에서 관리할 추가 열을 받고 있다.

```
export interface RegisteredClub {
  clubId: string;
  clubName: string;
  role: Role;
  moreColumns: {
    column: Column;
    value: String;
  }[];
  image: string;
  createdAt: Date;
  updatedAt: Date;
}

export type RegisteredClubs = Map<string, RegisteredClub>;
```

Fig. 8. 유저 객체에 들어가는 "가입된 동아리" 객체. 동아리 내의 다양한 정보를 담고 있으며 특히 moreColumns로 동아리의 추가 열과 그에 대한 답변을 담고 있다.

그렇기에 그림7과 같이 동아리 객체에 userColumns라는 feature를 받아 추가되는 열을 배열의 형태로 받았다. 그리고 유저 객체의 registeredClubs라는 열에 그림8과 같이 동아리 id(clubId)를 key, 동아리 정보를 value로 받은 Map(dictionary 자료형) 형태로 저장했다. 원래는 registeredClubs 안에 소속된 동아리 정보를 배열로 받았으나 코드의 가독성이 떨어지고 동아리를 찾는 데에 시간을 무의미하게 많이 소비하는 문제가 발생해 개발 도중 데이터 구조를 변경했다. 그림8의 가입한

동아리를 뜻하는 RegisteredClub 객체에는 moreColumns라는 열을 받아 동아리가 부원들에 한 질문(Column)과 그에 대한 답변(value:string)을 동시에 받았다.

User Column Hierarchy SKKUDO는 유저의 열을 마음대로 편집하는 기능은 기본적으로 그림9와 같은 아키텍처를 가진다. 예를 들어, 동아리에 "트랙" 열을 추가하게 되면 우선 Club의 정보를 수정하긴 하나 저장하지는 않는다. 그 이후 유저를 순회하면서 유저의 가입된 동아리(registeredClub) 정보를 변경하고 특별한 문제가 없으면 수정된 Club 객체를 저장한다. 수정과 삭제도 이와 동일한 방식으로 진행된다. 유저를 순회하면서 실행되는 구체적인 로직은 addColumn, updateColumn, deleteColumn과 같은 User의 method로써 구현된다.

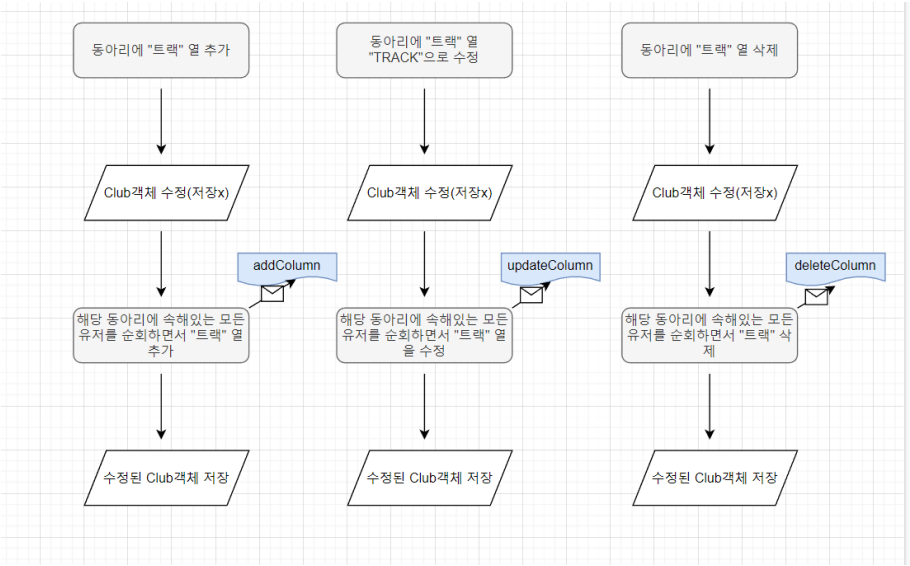

Fig. 9. 유저의 열을 편집하는 작업은 club 객체를 수정하고, 연관된 유저들을 모두 수정하는 식으로 이루어진다.

Validation Table SKKUDO에는 계속 언급했듯이 공지쓰기는 운영진 이상, 공지읽기는 부원 이상 등 기능마다 사용할 수 있는 권한이 정해져있다. 이를 구현하기 위해 동아리마다 Validation Table이라는 모델을 가진다. 이 Validation Table의 열은 기능이름(공지쓰기 등), 그에 상응하는 값은 권한(부원/운영진/부회장/회장)이다. 그림10와 같이 ValidationTable을 이용해서 프론트의 요청을 처리한다.이 과정에서 필요한 정보는 유저 정보를 가진 쿠키와 작업하려는 동아리의 id인데 프론트에서 자연스럽게 이것들을 보내주게 된다.

우선 findByClubId라는 User 객체의 메소드로 유저의 가입 동아리 정보를 가져온다. 만약 유저가 운영진이라는 권한을 가지고 있디먄, 이후 findValidator라는 Validation 객체의 메소드로 해당 요청의 유형을 자동으로 파악해 인증 테이블에서 해당 요청에 필요한 권한을 검색한다. 이 경우에는 공지 쓰기라는 요청이고, 이에

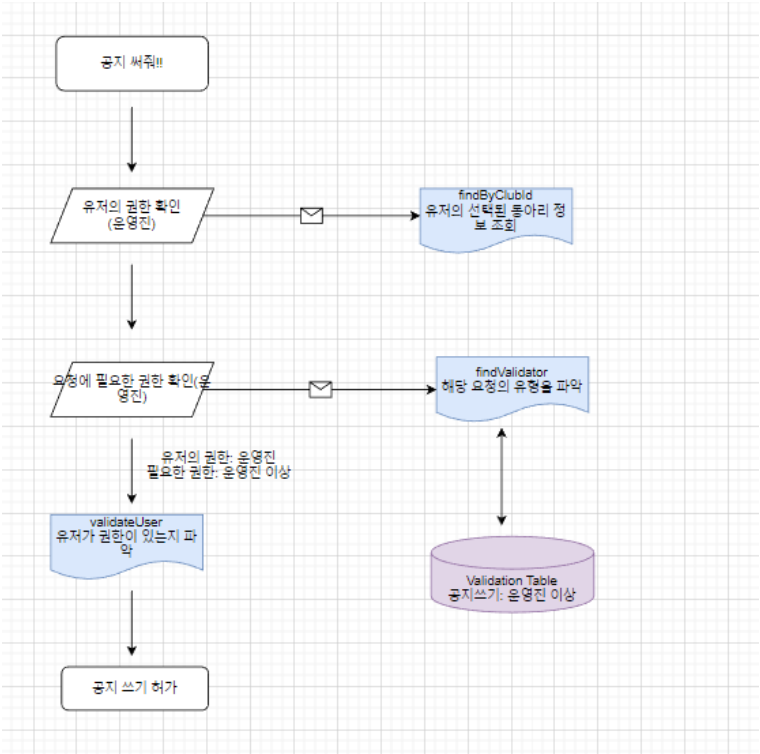

Fig. 10. 권한 테이블을 이용해 요청을 보낸 유저가 필요한 권한을 가지고 있는지 확인하는 과정이다.

필요한 권한은 운영진 이상이다. 마지막으로 ValidateUser라는 Validation 객체의 메소드로 유저의 권한(운영진)과 필요한 권한(운영진 이상)을 비교한다. 비교 결과 유저는 권한이 있는 것으로 파악되었으므로 공지를 쓰는 것을 허가한다. 만약 유저가 권한이 없다면 프론트에 403(Forbidden) response를 보낸다.

User Role Hierarchy 유저 권한을 수정할때 사용되는 로직은 다음과 같은 예시로 설명할 수 있다. 그림11은 그림10에 게시된 인증 테이블을 확인하는 과정을 전제한다. 또한 유저는 부회장이고, 유저 변경을 위한 권한은 운영진 이상인것을 가정하여 유저가 권한 변경이 가능한 사용자라는 것을 전제로 한다. 여기서 보아야할 것은 ”이 유저의 권한이 바꾸려는 유저의 권한보다 높은가”와 ”이 유저의 권한이 부여하려는 권한 이상”인가이다. 이를 위해 ValidateUser 메소드를 다시 호출하게 되는데, ”유저의 권한이 수정대상보다 높은가?”라는 질문에 답할 때는 유저 권한이 수정대상의 권한보다 높아도 에러를 반환한다.

그림11에서는 부회장이 부원을 바꾸기 때문에 ”내 권한이 수정대상 권한보다 높은가?” 질문에는 통과하지만, 회장 권한을 부여하려고 하기에 ”내 권한이 부여

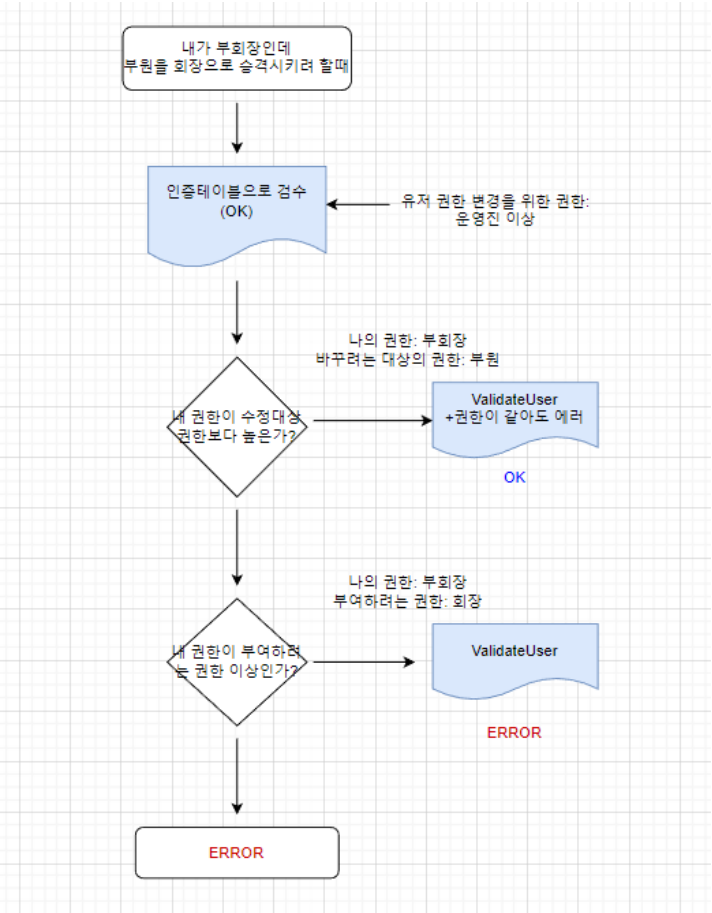

Fig. 11. 해당 유저가 다른 유저의 권한을 수정할 수 있는지 두 유저의 권한을 조회함으로써 파악한다.

하려는 권한 이상인가?"라는 질문에서 에러를 반환하게 된다.

Deploy SKKUDO의 배포는 신뢰성 있는 AWS EC2, MongoDB atlas, AWS Load Balancer로 이루어져 있다. 아래 그림12를 보면 유저는 www.skkudo.link 또는 api.skkudo.link로 요청을 보낸다. 이때 EC2 로드밸런서가 이를 각각 프론트엔드와 백엔드로 연결해준다. 프론트엔드 서버와 백엔드는 모두 하나의 EC2 인스턴스(서버 컴퓨터) 안에서 동작한다. 이 과정에서 로드밸런서에 SSL 인증서를 넣어주면

자동으로 https 설정이 가능하다. SSL 인증서를 발급받는 것은 AWS certificate manager 서비스를 이용했다. EC2 안의 백엔드 서버는 MongoDB atlas, 즉 외부 DB 서버와 소통한다.

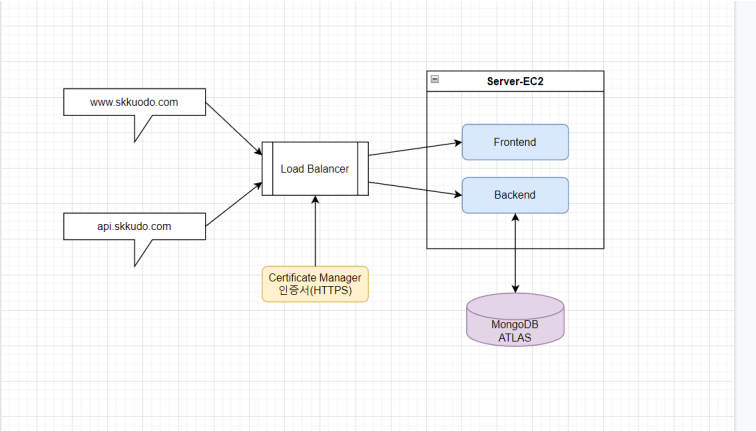

Fig. 12. www와 api 서브도메인으로 요청이 들어오면 각각 프론트엔드와 백엔드를 연결해주고, 이 과정에서 HTTPS 설정을 해준다. 백엔드는 외부 DB 서버인 MongoDB ATLAS와 소통한다.

4 Evaluation

SKKUDO는 타 플래폼과 차별화된 동아리 관리 서비스를 지원한다. 이 장에서는 어떠한 차별점을 가지고있는지 기술하고 차별점들을 평가한다.

공지 작성과 일정 작성과 같은 기본적인 기능은 물론이거니와 공지분류, 일정분류, 일정 겹치기와 같은 기능 또한 구현이 되어있다. 공지 및 일정 카테고리를 이용한 분류를 통해 공지 및 일정 검색을 원활하게 한다. 일정 겹치기의 경우 여러개의 일정이 겹치더라도 일정등록이 가능하게 하였다. 이를 통해 더 유연한 동아리 일정관리가 가능하다. 또한 일정 및 공지 숨기기 기능을 통해 일반 부원이 보아서는 안되는 공지/일정 또한 관리가 가능하다.

사용자가 원하는 서류/면접 질문을 등록할 수 있는 사용자 맞춤형 서비스를 제공한다. 최종 합격자 산출을 위해 서류/면접 점수 반영 비율을 등록할 수 있고, 자동으로 합격자 산출기능이 구현되어있다. 이 기능은 보다 빠르고 합리적인 모집관리를 도와준다.

동아리별로 유저 접근권한을 조정할 수 있다. 동아리 운영 중 가계부 관리, 동아리원 퇴출 등 민감한 사항에 대하여 권한이 있을 경우에만 접근할 수 있도록 하는 것은 매우 중요한 사안일 것이다.
또한 SKKUDO는 유저 관리 항목을 추가하여 관리할 수 있다. 유저의 정보를 추가 및 삭제가 가능하다. 이를 통해 유연하고 세심한 유저 정보 관리가 가능하다.

앞서 설명한 위 기능들을 베타 테스트를 통해 확인해 보았을 때 모든 기능이 문제없이 사용이 되는 것을 확인하였다. 위 기능들은 후술할 관련 타 동아리 플랫폼과 비교했을 때 SKKUDO만의 강점이라고 할 수 있으며 사용자는 이를 통해 SKKUDO가 더 나은 동아리 관리 플랫폼 및 앱이라는 것을 느낄 수 있다.

5 Limitations and Discussions

SKKUDO는 동아리 관리를 중점적으로 지원하는 플랫폼이다. 따라서 마이캠퍼스, SKKLUB등 타 플랫폼에 비해 동아리 소개페이지와 같은 동아리만의 홍보 기능은 지원하지 않는다. 따라서 동아리 일반부원 입장에서는 단지 공지 및 일정을 확인하는 용도로 사용될 수밖에 없다. SKKUDO에서의 더 좋은 사용자 경험과 동아리원 간의 원활한 커뮤니케이션을 위해, 향후 동아리 홍보뿐만 아니라 동아리 활동을 공유하고 소통할 수 있는 기능들 또한 필요하다면 추가하는 논의가 이뤄지고 있다. 현재 SKKLUB의 개발자와 두 플랫폼의 연계를 통한 더 큰 서비스로의 확장을 기획 중이다.

SKKUDO는 성균관대학교 교내 동아리들에 대하여 동아리 관리를 지원하는 플랫폼이다. 따라서 사용자들은 당연히 성균관대학교 구성원으로 제한된다. 이 때문에 회원가입 및 동아리 지원을 할 때 교내 구성원인지 확인하는 절차가 필요하다. 성균관대학교 메일 등을 이용하여 신원을 확인할 수 있는 방법이 있을 수 있으나, 시간적인 문제로 기능을 구현하지 못했다. 차후 개인정보 문제를 고려하여 성균관대학교 학교 메일을 통한 인증방식을 추가하여 더욱 안전한 SKKUDO의 운영을 계획하고 있다.

SKKUDO는 동아리 관리 플랫폼이므로 플랫폼의 안정성을 검증하기 위해서는 실제 동아리에서 운영을 해보아야 한다. 하지만 실제 동아리를 통한 베타 테스트 거치지 못했다는 한계점이 존재한다. 따라서 멋쟁이 사자처럼 이라는 교내 동아리를 대상으로 베타 테스트하기로 계약했으며, 6개월간의 테스트 이후 성균관대학교 학생들을 대상으로 정식 서비스를 실시할 전망이다.

6 Related work

동아리 관리 플랫폼은 SKKLUB, 동방, 마이캠퍼스, 네이버 카페 등 여러 서비스가 존재한다. 하지만 위와 같은 플랫폼들은 서류 및 면접 질문관리, 합격자 산출 등 동아리 관리 기능에 부실함을 보였다. 자세한 기능에 따른 비교는 정리해서 표1로 나타내었다.

표1에서 볼 수 있는것처럼, SKKUDO는 이전 서비스들과 차별화된 동아리 관리서비스를 지원하고 있다. SKKDO는 동아리의 목적에 맞게 커스터마이징이 가능하고, 사용하기 편리한 서비스를 제공한다. *'Customizable but Easy to Use'*는 SKKDO의 지향점이며 포지셔닝이라고 할 수 있다.

7 Conclusion

지금까지 성균관대학교의 동아리 및 학회의 단체 관리 시스템 SKKUDO에 대해 상세히 알아보았다. 프론트엔드는 성균관대학교를 상징하는 색의 조합으로 디자인되었으며, 접속한 사용자에 따라 페이지 구성이 나뉘고, 백엔드 서버와 Rest API

Table 1. 타 동아리 관련 앱들의 기능 비교 표

	SKKLUB	동방	마이캠퍼스	네이버카페
권한관리	X	X	O	X
일정관리	X	X	O	O
모집중인 동아리확인	X	X	O	X
서류/면접 질문관리	X	X	X	X
서류/면접 점수관리	X	X	X	X
합격자 자동산출	X	X	X	X
서류/면접 반영비율	X	X	X	X
관리항목 추가	X	X	X	X

방식을 통해 데이터를 주고받는다. 백엔드는 동아리 관리에 필수적인 모집 관리, 일정 관리, 공지 관리, 유저 관리, 권한 관리 등의 기능을 구현하여 마찬가지로 API를 통해 프론트엔드와 데이터를 주고받으며, SKKUDO의 핵심 기능이자 차별점인 유저 열 커스터마이징 기능을 구현하기 위해 자유도가 높은 비관계형 데이터베이스인 MongoDB를 채택하였다. 또한 bcrypt 암호화 알고리즘을 통해 비밀번호를 암호화하고, AWS EC2의 로드밸런서 서비스를 통해 https를 설정함으로써 인터넷 보안 문제에 관해서도 대비책을 강구하였다. 마지막으로 Validation Table을 작성하여 사용자 사이의 계층 구조를 조직하였으며, SKKUDO의 다양한 페이지와 기능을 각 사용자에게 올바르게 제공하도록 만들 수 있었다.

이제 여러분은 SKKUDO가 어떻게 탄생하게 되었으며, 어떻게 구성되었고, 어떤 기능이 있으며, 어떻게 완성되었는지 알게 되었다. 여태까지 사용자가 완벽히 만족할 수 있는 동아리 관리 플랫폼이는 존재하지 않았고, 그렇기 때문에 더더욱 SKKUDO가 정식 서비스 중에도 제 기능을 톡톡히 해낼 수 있을 것으로 전망한다. 앞서 언급했던 것처럼 SKKUDO는 약 반년이라는 테스트 기간을 거친 후 성균관대학교 학생들을 대상으로 정식 서비스를 실시할 예정이며, 우리는 SKKUDO가 많은 대학생의 동아리 활동을 보다 더 편리하고 행복하게 만들 수 있기를 기대한다.

References

1. React Homepage, https://ko.reactjs.org/. Last accessed 27 Nov 2022
2. Style components Homepage, https://styled-components.com/. Last accessed 27 Nov 2022
3. Axios Docs, https://axios-http.com/kr/docs/intro. Last accessed 27 Nov 2022
4. TanStack React Query Docs, https://tanstack.com/query/v4/docs/overview. Last accessed 27 Nov 2022
5. React Router Dom Homepage https://reactrouter.com/en/main. Last accessed 27 Nov 2022
6. React Icons Homepage https://react-icons.github.io/react-icons/. Last accessed 27 Nov 2022
7. Npm react calendar page https://www.npmjs.com/package/react-calendar. Last accessed 27 Nov 2022
8. Apex Chart Homepage https://apexcharts.com/. Last accessed 27 Nov 2022
9. MUI Homepage https://mui.com/. Last accessed 27 Nov 2022
10. Framer Motion Homepage https://www.framer.com/motion/. Last accessed 27 Nov 2022
11. Recoil Homepage https://recoiljs.org/ko/. Last accessed 27 Nov 2022
12. NodeJS Homepage https://nodejs.org/ko/. Last accessed 27 Nov 2022
13. Express Homepage https://expressjs.com/. Last accessed 27 Nov 2022
14. Npm Nodemonpage https://www.npmjs.com/package/nodemon. Last accessed 27 Nov 2022
15. Mongoose Homepage https://www.npmjs.com/package/nodemon. Last accessed 27 Nov 2022
16. Express CORS Docs https://expressjs.com/en/resources/middleware/cors.html. Last accessed 27 Nov 2022
17. Npm Cookie Parser Page https://www.npmjs.com/package/cookie-parser. Last accessed 27 Nov 2022
18. Npm Dotenv Page npmjs.com/package/dotenv. Last accessed 27 Nov 2022
19. Npm Bcrypt Page https://www.npmjs.com/package/bcrypt. Last accessed 27 Nov 2022
20. Npm Multer Page https://www.npmjs.com/package/multer. Last accessed 27 Nov 2022