축구 데이터 운용 웹 서비스 개발

조규철*, 이해찬O
*인하공업전문대학 컴퓨터정보공학과,
O인하공업전문대학 컴퓨터정보공학과
e-mail: kccho@inhatc.ac.kr*, gisoun98@gmail.comO

Development of Web Application Service to Football Data Management

Cho Kyu Cheol*, Lee Hae ChanO
*Dept. of Computer Science & Engineering, Inha Technical College,
ODept. of Computer Science & Engineering, Inha Technical College

● 요 약 ●

본 논문에서는 Vue.js와 Spring Boot 프레임워크를 사용하여 사용자 편의성을 제공하기 위해 Gird System, Data Table, Chart 등의 기능을 활용하여 축구 데이터를 표현한 웹 응용서비스를 개발하였다. Rapid API의 Open REST API 중 하나인 API-FOOTBALL에 액세스하여 국내에서 쉽게 찾아보기 어려운 세계 축구 리그의 순위표, 일정, 팀 및 선수의 세부 정보 데이터를 활용하였다.

키워드: 축구(football), Web Application Service, Vue.js, Spring Boot, REST API

I. Introduction

2021년, 문화체육관광부에서 발표한 '주요 프로스포츠 경기당 평균 관중 추이'[1]에 따르면 축구 종목의 평균 관중 추이는 1,382명으로 평균 관중 1,937명인 야구 종목에 이은 2위를 차지하였다.

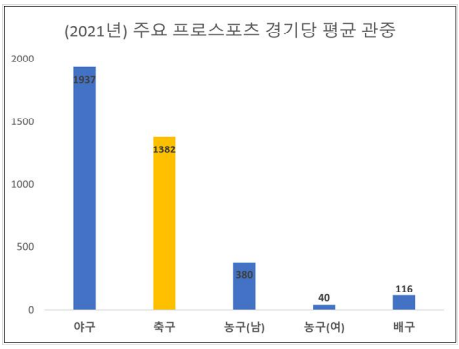

Fig. 1. average crowd per major professional sports match[1]

이는 국내에서 축구 종목이 사람들에게 큰 인기를 얻고 있음을 보여준다.

축구 종목의 인기가 높아지는 만큼 자신이 응원하는 팀과 선수의 정보, 기록 등을 쉽게 확인할 수 있는 Web Application Service(이하, WAS)의 필요성 또한 높아지고 있다. 하지만 국내에는 이러한 정보를 제공하는 서비스가 많지 않다. 특히나 해외의 축구 리그 정보의 경우 세계적으로 큰 인지도가 있는 소수의 유럽 내 축구 리그만을 다루기에 얻을 수 있는 정보의 범위 또한 매우 좁다. 국내에서는 대부분 해외의 WAS에 접근하여야 원하는 정보를 얻을 수 있다. 그러나 국내 사용자로서 해외 WAS의 경우 대부분 해외에 서버를 두고 운영하기에 국내에서 접근하였을 때 해외에서보다 상대적으로 낮은 성능을 보인다.

본 연구는 위와 같은 문제점에서 착안하여 국내·외의 축구 데이터를 운용할 수 있는 WAS를 효율적인 Grid System 및 범용성이 뛰어난 전용 라이브러리들을 사용할 수 있는 Vue.js와 내장 웹 서버를 제공하는 MVC 패턴 기반의 Spring Boot 기반으로 개발하는 것에 목적을 두어 진행하였다.

II. Data Collection

본 연구를 진행하기에 앞서 프로 축구 리그가 존재하는 국가들의 데이터를 수집하였다. 대규모 API 마켓플레이스인 Rapid API에서 API-FOOTBALL[2]을 사용하였다. 해당 API를 통해 950개 이상의 리그 및 컵 대회, 선수, 순위 등 방대한 정보를 오브젝트 형태로 얻을 수 있는 API 엔드포인트에 액세스하였다.

액세스는 파이썬 언어 환경에서 진행되었다. API- FOOTBALL에서 주어지는 고유한 URL과 API-KEY, 그리고 필요에 따라 파라미터

를 조정하고 requests 라이브러리를 통해 API 응답을 요청하였다. 정상적으로 응답을 받았다면 json 모듈을 사용하여 오브젝트 형태의 응답을 JSON 파일로 변환시킨 후 지정된 로컬 경로에 저장한다. 이러한 과정은 아래 그림2에서 확인할 수 있다.

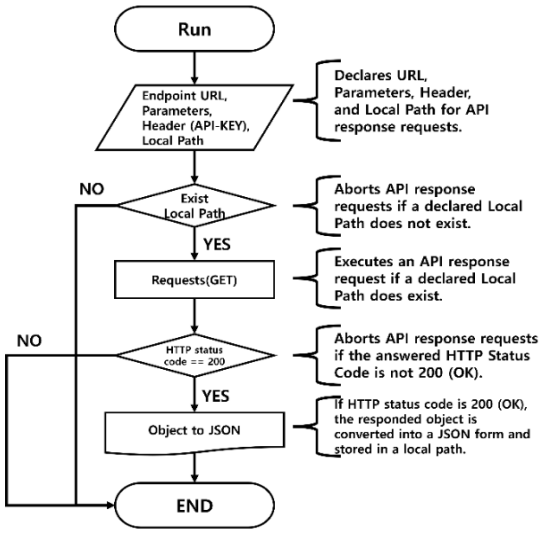

Fig. 2. API Endpoint Access Process Flowchart

수집된 데이터는 Front-End의 Axios 인스턴스에 의해 실행된 Back-End 컨트롤러에서 ObjectMapper를 통해 Java 객체로 사용된다. 아래 그림3은 본 연구에서 사용된 전 세계 리그 데이터가 담겨있는 JSON 파일 데이터 일부이다.

```
"response": [
    {
        "league": {
            "id": 62,
            "name": "Ligue 2",
            "type": "League",
            "logo": "https://media.api-sports.io/football/leagues/62.png"
```

Fig. 3. Part of JSON File Data

III. Development

1. Front-End

1.1 System Environment

Front-End에서의 개발환경은 아래 표 1과 같이 구축하였다.

Table 1. Front-End System Environment

Class	Name	Version
OS	Windows 10	64bit, 21H2
Framework	Vue.js	Vue 3
Runtime	Visual Studio Code	1.71.0
API	Vuetify	3.0.0-beta.0
	Vue Router	4.0.3
	Pinia	2.0.22
	Axios	0.27.2
	Vue3 Easy Data Table	1.4.19
	Day.js	1.11.6
	V Calendar	3.0.0-alpha.8
	Chart.js	3.9.1

1.2 Routing

SPA 관점에서의 Routing은 View를 전환하는 내비게이션을 관리하기 위한 기능을 의미한다. 본 연구에서 사용된 Vue Router는 Vue.js에서 페이지 간 이동을 위해 쓰이는 라이브러리로, Vue.js의 공식 Router이다.

페이지 이동 시 Vue Router에 URL과 페이지 용도에 맞는 파라미터를 push하여 전달한다.

```
path: "/competitions/:country/:id/:season",
name: "Competitions",
component: () =>
  import("@/views/competitions/CompetitionsView.vue"),
props: true,
```

Fig. 4. Part of routing information on the page

위 그림4는 페이지의 라우팅 정보의 일부이다. 위와 같이 competitions 경로 뒤에 country, id, season 파라미터를 받기에 같은 컴포넌트임에도 각각 다른 데이터를 표현하는 것이 가능하다.

1.3 Grid System

질서있는 구조의 웹 사이트 개발을 위해서는 애플리케이션 콘텐츠 내에 특정 레이아웃을 구성할 수 있는 Grid System을 활용할 수 있다. 본 연구에서는 Vue.js에서 제공하는 UI 프레임워크인 Vuetify의 Grid System[3]을 활용하여 레이아웃을 구성하였다. Vuetify의 Grid System은 아래의 그림5와 같이 가장 큰 틀인 v-container 안에 v-row, v-col 순으로 구성되어있으며 하나의 열에 12개의 행이 포함된다.

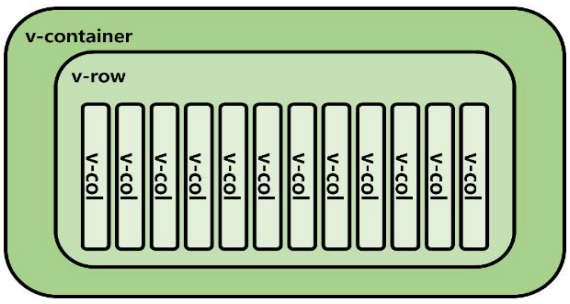

Fig. 5. Vuetify Grid System

2. Back-End

2.1 System Environment

Back-End에서의 개발환경은 아래 표 2와 같이 구축하였다.

Table 2. Back-End System Environment

Class	Name	Version
OS	Windows 10	64bit, 21H2
Framework	Spring Boot	2.7.3
Server	Apache Tomcat	Tomcat 9
IDE	Visual Studio Code	1.71.0

2.2 VO(Value Object)

본 연구의 Back-End에서는 데이터 대부분을 프로퍼티가 지정되어 있는 JSON 파일로 다루기에 JSON의 각 객체명 및 필드명으로 구성된 VO Class를 작성하였다. 이때, JSON 파일의 불필요한 데이터 읽기를 방지하기 위해 모든 요소의 Getter/Setter를 VO에 작성하지 않고 사용할 요소의 객체 또는 필드의 Getter/Setter를 작성하였다. 아래의 그림6은 사용된 VO Class의 일부이다.

```
public class GoalsVO {

    @JsonProperty("for")
    private int scored;
    private int against;

    public int getScored() {
        return scored;
    }

    public void setScored(int scored) {
        this.scored = scored;
    }

    public int getAgainst() {
        return against;
    }
}
```

Fig. 6. Part of VO Class used

2.3 Controller

MVC 패턴에서 Controller는 사용자의 요청을 받아 오브젝트를 전달하는 중요한 역할을 한다. 본 연구에서는 JSON 형태의 데이터를 오브젝트 데이터로 반환받을 수 있는 RestController 어노테이션을 사용하였다. 그리고 로컬 경로에 저장된 JSON 파일을 읽어온 뒤 Jackson 라이브러리의 ObjectMapper를 통해 VO Class를 매핑하여 필요한 데이터를 오브젝트 형태로 반환하였다.

3. Key Features

3.1 App Header

App Header는 어떤 페이지로 가도 항상 고정된 위치에 존재하기에 모든 페이지에서 사용할 수 있는 Header 컴포넌트이다. App Header에는 '대회' 탭이 존재하며 해당 탭에 마우스 오버 시 A-Z로 시작하는 국가명 서브 메뉴를 확인할 수 있다. 국가명을 클릭할 시 해당 국가에 존재하는 프로 축구 리그 리스트가 나타난다.

리그를 클릭하면 /competitions/ URL에 국가명, 리그 ID, 현재 시즌을 Parameters로 하고 Vue Router에 Push 하여 클릭한 리그의 순위 테이블(League Standings), 일정(League Fixtures)을 확인할 수 있는 페이지로 이동한다.

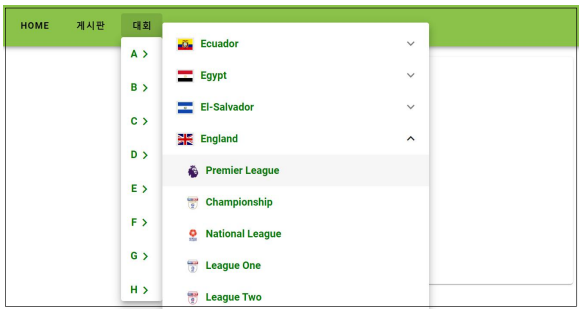

Fig. 7. App Header

3.2 Competitions

/competitions/ 페이지에서는 리그의 순위 테이블과 리그 일정을 확인할 수 있는 탭이 존재한다. 리그의 순위 테이블은 Vue 3 버전 기반의 커스텀 데이터 테이블인 Vue3 Easy Data Table로 제작하였다. Back-End와의 통신을 통해 반환받은 리그 데이터를 기반으로 테이블의 헤더와 구성 요소를 Template에 담아 표현하였다.

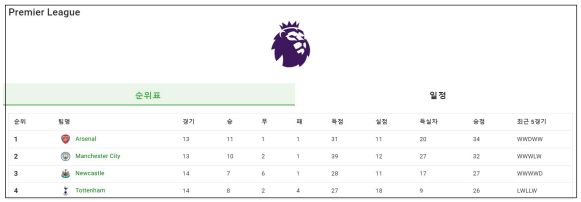

Fig. 8. League Standings

'일정' 탭에서는 리그의 일정 데이터를 기반으로 제작되었으며, (N, 5) 행렬로 구성된 Grid 형태로 이루어져 있다. 이곳에서의 날짜, 시간 데이터 처리는 Day.js 라이브러리를 사용하였다. 또한, V Calendar 라이브러리를 사용하여 날짜를 클릭하면 Date Picker가 나타나고 직접 경기 일정이 있는 날짜를 선택할 수 있다. 이때, 경기 일정이 없는 날짜의 경우 비활성화하여 표현을 생략하였다.

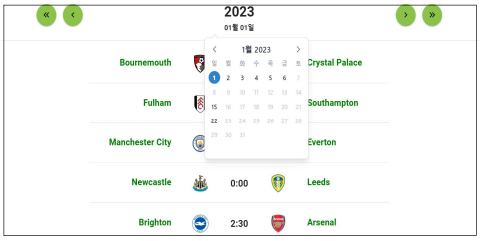

Fig. 9. League Fixtures

해당 페이지에서 팀명을 클릭하면 /teams/ URL에 국가명, 리그 ID, 팀 ID를 Parameters로 하고 Vue Router에 Push 하여 클릭한 팀의 세부 정보(Team Detail)를 확인할 수 있는 페이지로 이동한다.

3.3 Team Detail

/teams/ 페이지에서는 간략히 확인할 수 있는 팀의 순위 테이블과 리그 일정, 역대 리그 순위 차트, 스쿼드 컴포넌트로 구성되어있다. 팀의 순위 테이블 및 리그 일정 컴포넌트는 /competitions/ 페이지에서 데이터를 받아와 표현하였다.

Fig. 10. Standings and Fixtures of Team

팀의 역대 리그 순위를 선형 차트로 표현하기 위해서 Chart.js 라이브러리를 사용하였다. Chart.js의 Custom Plugins 기능과 상태 관리 라이브러리 Pinia를 활용하여 팀의 하부/상위리그의 순위를 반영하였다.

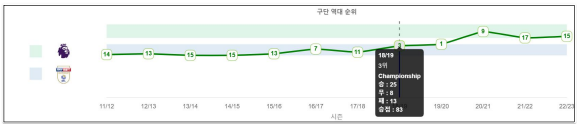

Fig. 11. Team's rankings of all-time

팀의 현재 스쿼드는 각 포지션별 (N, 3) 행렬로 구성된 Grid 형태로 표현하였다.

Fig. 12. Squad of Team

스쿼드 컴포넌트의 선수를 클릭하면 /players/ URL에 선수 ID를 Parameters로 하고 Vue Router에 Push 하여 클릭한 선수 세부 정보(Player Detail)를 확인할 수 있는 페이지로 이동한다.

3.4 Player Detail

/players/ 페이지에서 좌측 선수의 프로필 사진에는 국적, 포지션, 소속팀 등번호가 함께 나타나며, 우측에는 현 소속팀, 선수명, 생년월일, 키 및 몸무게가 나타난다. 하단에는 시즌을 기준으로 내림차순 정렬된 선수의 역대 커리어를 테이블 형태로 표현하였다.

Fig. 13. Player Details

IV. Conclusions

본 연구는 국내에서 다양한 국가의 축구 데이터를 쉽게 확인하는 것이 어렵다는 문제점에서 착안하여 국내·외의 축구 데이터를 운용할 수 있는 SPA 기반 WAS 개발에 목적을 두었다. 본 연구를 통해 국내에서 다양한 국가의 축구 데이터를 쉽게 확인하여 국내 사용자들에게 편의성을 줄 것으로 기대한다.

REFERENCES

[1] 문화체육관광부, 주요 프로스포츠 경기당 평균 관중 추이, https://www.index.go.kr/potal/main/EachDtlPageDetail.do?idx_cd=1662#quick_01

[2] Rapid API, API-FOOTBALL, https://rapidapi.com/api-sports/api/api-football

[3] Vuetify, Gird System, https://next.vuetifyjs.com/en/components/grids/