**Aprosolution 반응형 웹개발 개발 표준 정의서**

**1. 개요 및 목적**

**- 프로젝트 개요**

- 프로젝트 이름 : 'Aprosolution'

**- 요구페이지**

- Main Page

- COMPANY [ 회사소개, 인재채용, 공지사항, 인트라넷 ]

- BUSINESS [ 개요, 컨설팅, 서비스, 솔루션 ]

- PROJECT [ 프로젝트 실적 ]

- CONTACT [ 문의사항, 문의사항내용 ]

**- 개발 목표**

- 위 명시된 참고 페이지를 틀로 반응형 웹페이지를 제작하여 퍼블리셔 및 프론트엔드와의 협업을 할 수 있도록 웹페이지 작업

**- 화면 구성 및 하위 페이지**

**1. Main Page**

**2. Company**

* 회사 소개
* 인재 채용
* 공지사항
* 인트라넷

**3. 인트라넷**

* **로그인**
* **Admin Main Page**
* **직원 관리**
  + 직원 리스트
  + 직원 상세
  + 직원 추가
  + 직원 투입 현황
* **게시글 관리**
  + **공지사항**
    - 공지사항 리스트
    - 공지사항 상세
  + **프로젝트**
    - 프로젝트 리스트
    - 프로젝트 상세
    - 직원 찾기
  + **프로젝트 실적**
    - 프로젝트 실적 리스트
    - 프로젝트 실적 상세
  + **문의사항**
    - 문의사항 리스트
    - 문의사항 상세
* **자료실**
  + 자료실 리스트
  + 자료실 상세
* **공통코드 관리**
  + 공통 그룹코드 및 상세 코드 리스트
  + 공통 그룹코드 상세
  + 공통 그룹코드 등록

**4. Business**

* 개요
* 컨설팅
* 서비스
* 솔루션

**5. Project**

* **프로젝트 실적**
  + 프로젝트 실적 리스트
  + 프로젝트 실적 등록 페이지
  + 프로젝트 실적 수정 페이지

**6. Contact**

* 문의사항 등록
* 문의사항 리스트
* 문의사항 상세 및 수정 페이지

**2. 개발 환경**

**- 개발 환경 및 Tool 버전**

- 개발 언어 : JAVA 17

- 데이터베이스 언어 : MySQL

- Server : Apatche Tomcat 10.1.31

- Framework : Spring Boot 3.4.0, React

- 형상 관리 도구 : Github

- Build Tool : Gradle-8.10.2

- IDE Tool

- FrontEnd : Visual Studio Code

- BackEnd : STS4

- DataBase : MySQLWorkBench

**- 개발환경 결정 사유**

- **Spring boot 3.4.0, Java 17** - Spring boot는 기존 Spring boot 2버전을 서비스 종료하고 Spring boot 3버전대를 지원하며 최신버전 기준 안정화 버전이 3.4.0으로 결정 - Spring boot 3버전부터 지원되는 JAVA의 지원 버전은 JAVA 17버전부터 지원을 하며 LTS 버전인 JAVA17버전을 채택

- **Apatche Tomcat 10.1.31, STS4** - Apatche Tomcat 10버전부터 기존 Java EE가 Jakarta EE로 명칭이 바뀌면서 Java EE(javax) 기반에서 Jakarta EE기반 으로 되었다. - 그러나 Apatche Tomcat 공식 홈페이지에서 10.0의 버전이 더이상 지원되지 않고 10.1버전을 지원한다. - Java의 LTS 버전인 8버전과 11버전은 기존 Apatche Tomcat 9버전을 지원하지만 Java 17버전은 9버전과 호환되지 않는다. - Eclipse와 STS3의 경우 Apatch Tomcat 10.1의 호환이 안되는 사항이 발생하여 Apatche Tomcat 10.1을 지원하는 STS4를 채택하게 되었다.

- **Visual Studio Code** 각종 add-on 도구 포함 및 관련 및 터미널을 프로그램 내에서 사용할 수 있는 점과 React와 같은 프레임워크에 대해 최적화되어 프론트 IDE 도구로 채택

**- MySQL** - 기존 Oracle DB의 경우 개인, 개발, 테스트, 데모 용도로는 무료로 사용할 수 있으나 상업적 또는 운영 환경에서 사용하려면 유료로 전환이 되어 별도 구매가 필요하다. 이에 MySQL을 채택

**3. 프론트엔드 개발 표준**

**- React 컴포넌트 선언 방식**

- 컴포넌트 선언 방식 중 클래스형 선언, 함수형 선언 중 함수형 선언 방식을 채택

**- React Compnant 명명 규칙**

- 컴포넌트를 생성할 시 반드시 가장 첫 글자는 대문자로 만들어야하며 폴더는 소문자로 시작하는 카멜케이스로 작성해야한다. - 컴포넌트를 만드는 파일은 대문자로 시작하는 카멜케이스로 이름을 지어야한다.

**- VSCode 디렉토리 구조 규정**

├── public/

│ ├── assets/css # 퍼블리싱을 위한 css 폴더

│ ├── html # 퍼블리싱을 위한 html 폴더

│ ├── images/ # 이미지 파일을 위한 폴더

│ ├── icons/ # 아이콘 파일을 위한 폴더

│ ├── index.html # SPA의 진입 HTML 파일

│ ├── favicon.ico # 파비콘 아이콘

├── src/ │

│ ├── assets/ # 스타일 및 이미지 파일 등 공용 자산

│ ├── css/ # 리액트 css 보관 파일

│ ├── JS/ # 리액트 JavaScript 보관 파일

│ ├── components/ # 재사용 가능한 컴포넌트들

│ ├── pages/ # 각 페이지 컴포넌트

│ ├── hooks/ # 커스텀 Hook 저장

│ ├── services/ # API 요청과 관련된 파일들

│ ├── App.js # 루트 컴포넌트

│ ├── index.js # 렌더링과 ReactDOM 설정

│ └── App.css # 전역 스타일 파일

**4. 백엔드 개발 표준**

**-변수 및 테이블 명명 규칙**

- 클래스 및 Exception : PascalCase

- 함수 및 메서드, 변수 : camleCase

- 상수 : UPPERCASE

- DB 테이블명, 필드명 : SNAKE\_CASE

**- Spring Boot 디렉토리 구조 규정**

project-root

├── src

│ ├── main

│ │ ├── java

│ │ │ └── com

│ │ │ └── Aprosolution

│ │ │ ├── config # Spring 설정 파일 (MyBatis, Security, Swagger 등)

│ │ │ ├── controller # REST API 컨트롤러

│ │ │ ├── domain # Entity, VO 등 도메인 클래스

│ │ │ ├── dto # 요청/응답 DTO 클래스

│ │ │ ├── exception # 예외 처리 관련 클래스

│ │ │ ├── mapper # MyBatis Mapper 인터페이스

│ │ │ ├── repository # MyBatis Repository (매퍼 XML 파일과 매핑됨)

│ │ │ ├── service # 비즈니스 로직 서비스 계층

│ │ │ └── util # 유틸리티 클래스

│ │ └── resources

│ │ ├── mapper # MyBatis Mapper XML 파일

│ │ ├── static # 정적 리소스 (CSS, JS, 이미지 등)

│ │ ├── templates # JSP 또는 Thymeleaf 템플릿 (필요 시)

│ │ ├── application.properties # Spring Boot 설정 파일

└── ...

**5. 코드 관리 및 협업 기준**

**- Front영역 Back영역의 형상관리 구분 및 프로젝트 이름**

**- Front 영역 형상관리 프로젝트 정보**

**- Repository 이름** : Apro

**- 생성자 및 관리자** : 임현우

**- 브랜치 종류 :**

- main[master]

- feature/dev/{개발자명}

- 각 브랜치는 feature/dev/{개발자명}으로 작성하도록 명시

**- Back 영역 형상관리 프로젝트 정보**

**- Repository 이름** : Apro\_server

**- 생성자 및 관리자** : 임현우

**- 브랜치 종류 :**

- main[master]

- feature/dev/{개발자명}

- 각 브랜치는 feature/dev/{개발자명}으로 작성하도록 명시

**- Commit & merge 기준**

1) 프로젝트의 수정사항은 자신의 Branch에 commit을 시킨다.

2) Merge에 대한 Request를 생성 후 Main에 Merge

3) Merge 이후 팀원들은 프로젝트 수정사항은 Main[master]에서 pull을 받아 자신의 프로젝트에 적용 후 에러 사항을 항시 체크