WEEK3 - API 가이드 문서 작성

0. 개발 환경 비교

항목	스프링(Spring)	스프링 부트(Spring Boot)
설정 복잡도	XML 설정 파일을 사용하여 수동 설정	`application.yml` 또는 `application.properties` 파일로 간단히 관리
서버 설정	외부 서버(Tomcat 등) 설치 후 애플리케이션 배포	내장 서버(Tomcat 등) 지원, `java -jar`로 바로 실행 가능
의존성 관리	필요한 라이브러리를 수동으로 추가 및 버전 관리	Spring Boot Starter로 의존성 자동 관리 및 호환성 보장
초기 개발 속도	설정이 많아 시간이 걸리며 기본 설정 지원 없음	자동 설정 및 기본 설정 제공으로 빠르게 개발 시작 가능
생산성	설정과 서버 관리에 시간이 소요될 수 있음	설정과 실행이 간단해 개발 생산성이 높음

1. Rest API 학습

1.1 HTTP 통신 개요

HTTP 통신이란

- HTTP (HyperText Transfer Protocol)는 웹 브라우저와 서버 사이에서 데이터를 주고받기 위한 기본 프로토콜.
- **클라이언트-서버 모델**을 기반으로, 클라이언트가 요청(Request)을 보내고 서버가 응답(Response)을 반환하는 구조.
- 주로 **GET, POST, PUT, DELETE** 같은 HTTP 메서드를 사용해 작업을 수행.

1.2 HTTP 통신 주요 요소

1. **HTTP 요청**: 클라이언트가 서버에 보내는 요청. 주요 구성 요소:
    - **HTTP 메서드**: 요청 종류를 지정 (예: `GET`, `POST`, `PUT`, `DELETE`)
    - **URL**: 요청 대상 리소스를 지정
    - **요청 헤더**: 요청에 대한 메타 정보

- **요청 본문**: 필요한 데이터 (POST, PUT에서 사용)

2. **HTTP 응답**: 서버가 요청에 대한 결과를 반환. 주요 구성 요소:
   - **상태 코드**: 요청 처리 결과 (`200` 성공, `404` 리소스 없음, `500` 서버 오류)
   - **응답 헤더**: 응답에 대한 메타 정보
   - **응답 본문**: 서버가 클라이언트에 전송할 데이터

1.3 URL 요청 후 서버 응답 과정

1. URL 입력
2. DNS 조회
3. TCP 연결
4. HTTP 요청 전송
5. 서버에서 요청 처리
6. HTTP 응답 전송
7. 브라우저가 응답 처리

---

2. 인터페이스 가이드 문서 작성

2.1 API 개요

API 명칭: SW 활용 현황 통계 API

기능:

- 2020년도 총 로그인 수, 월별 및 일자별 접속자 수, 평균 하루 로그인 수, 휴일을 포함한 로그인 수, 부서별 월별 로그인 수 등의 통계 데이터를 제공.

2.2 개정 이력

버전	변경일	변경 사유	변경 내역
1.0	2024-10-22	최초 작성	SW 활용률 통계 API 최초 작성

---

3. API 목록

엔드포인트	설명	HTTP 메서드
`/api/total-logins-2020`	2020년도 총 로그인 수 조회	GET

`/api/monthly-user-counts`	월별 접속자 수 조회	GET
`/api/daily-user-counts`	일자별 접속자 수 조회	GET
`/api/average-daily-logins`	평균 하루 로그인 수 조회	GET
`/api/total-logins-including-holidays`	휴일을 포함한 로그인 수 조회	GET
`/api/department-monthly-login-counts`	부서별 월별 로그인 수 조회	GET

4. API 상세 설명

4.1 2020년도 총 로그인 수 조회

- **엔드포인트**: `/api/total-logins-2020`
- **HTTP 메서드**: GET
- **설명**: 2020년도 총 로그인 수를 반환.
- **응답 예시**:

```
{
  "totalLogins": 12345,
  "status": "success"
}
```

4.2 월별 접속자 수 조회

- **엔드포인트**: `/api/monthly-user-counts`
- **HTTP 메서드**: GET
- **설명**: 월별 접속자 수를 반환.
- **응답 예시**:

```
{
  "monthlyUserCounts": [
    {
      "month": "2020-01",
      "monthlyUserCount": 123
    },
    {
      "month": "2020-02",
```

```
      "monthlyUserCount": 456
    }
  ],
  "status": "success"
}
```

4.3 일자별 접속자 수 조회

- **엔드포인트**: `/api/daily-user-counts`
- **HTTP 메서드**: GET
- **설명**: 일자별 접속자 수를 반환.
- **응답 예시**:

```
{
  "dailyUserCounts": [
    {
      "date": "2020-01-01",
      "dailyUserCount": 10
    },
    {
      "date": "2020-01-02",
      "dailyUserCount": 15
    }
  ],
  "status": "success"
}
```

4.4 평균 하루 로그인 수 조회

- **엔드포인트**: `/api/average-daily-logins`
- **HTTP 메서드**: GET
- **설명**: 평균 하루 로그인 수를 반환.
- **응답 예시**:

```
{
  "averageDailyLogins": 50.25,
```

```
  "status": "success"
}
```

4.5 휴일을 포함한 로그인 수 조회

- **엔드포인트**: `/api/total-logins-including-holidays`
- **HTTP 메서드**: GET
- **설명**: 휴일을 포함한 2020년도 총 로그인 수를 반환.
- **응답 예시**:

```
{
  "totalLoginsIncludingHolidays": 18000,
  "status": "success"
}
```

4.6 부서별 월별 로그인 수 조회

- **엔드포인트**: `/api/department-monthly-login-counts`
- **HTTP 메서드**: GET
- **설명**: 부서별 월별 로그인 수를 반환.
- **응답 예시**:

```
{
  "departmentMonthlyLoginCounts": [
    {
      "departmentId": "IT",
      "month": "2020-01",
      "monthlyLoginCount": 250
    },
    {
      "departmentId": "HR",
      "month": "2020-01",
      "monthlyLoginCount": 100
    }
  ],
```

```
  "status": "success"
}
```

---

5. SQL 쿼리 설명 (통계 API 구현용)

5.1 월별 접속자 수

```
SELECT SUBSTR(createDate, 1, 6) AS month, COUNT(DISTINCT us
erID) AS monthly_user_count
FROM statistic.requestInfo
GROUP BY month
ORDER BY month;
```

5.2 일자별 접속자 수

```
SELECT SUBSTR(createDate, 1, 8) AS date, COUNT(DISTINCT use
rID) AS daily_user_count
FROM statistic.requestInfo
GROUP BY date
ORDER BY date;
```

5.3 평균 하루 로그인 수

```
SELECT AVG(daily_login_count) AS average_daily_logins
FROM (SELECT SUBSTR(createDate, 1, 8) AS date, COUNT(userI
D) AS daily_login_count FROM statistic.requestInfo GROUP BY
date) AS daily_counts;
```

5.4 휴일을 포함한 총 로그인 수

```
SELECT COUNT(userID) AS total_logins FROM statistic.request
Info WHERE createDate LIKE '20%';
```

5.5 부서별 월별 로그인 수

```
SELECT u.HR_ORGAN AS departmentId, SUBSTR(r.createDate, 1,
6) AS month, COUNT(r.userID) AS monthly_login_count
FROM statistic.requestInfo r
JOIN statistic.user u ON r.userID = u.userID
GROUP BY departmentId, month
ORDER BY departmentId, month;
```