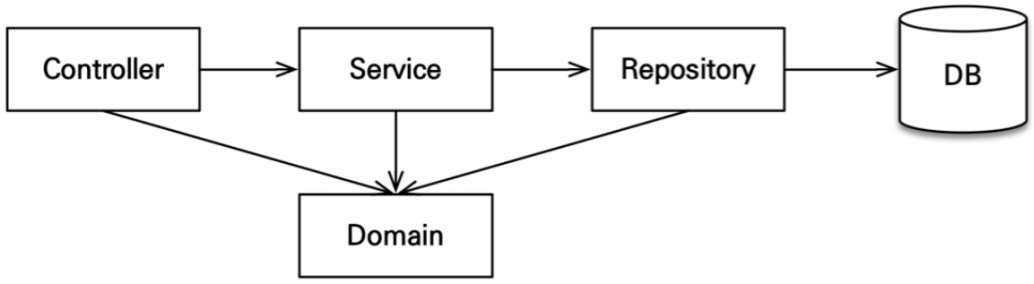
설문지 프로젝트 Spring Boot

요구사항

버전정보

- Java 17
- Spring Boot 3.1.5
- JPA
- H2 DB 2.2.224

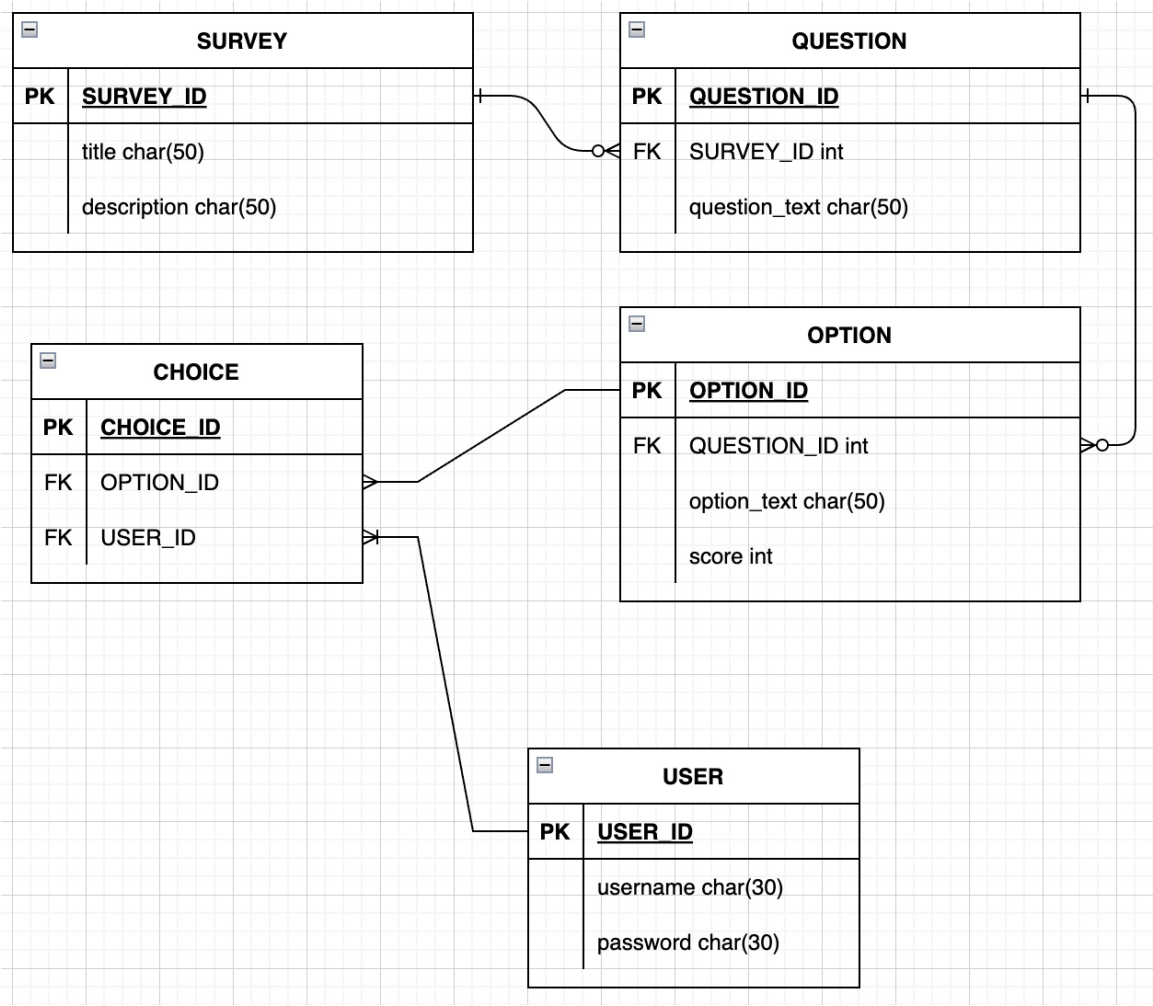
애플리케이션 아키텍쳐



- 계층형 구조 사용
  - Controller: RESTful API 구현
  - Service: 비즈니스 로직, 트랜잭션 처리
  - Repository: JPA 직접 사용하는 계층
  - Domain: 엔티티 모여있는 계층

객관식 설문지의 데이터 베이스 설계

데이터베이스 ERD



연관관계

Survey(설문지)

- 여러 Questions 가질 수 있음

Question(문항)

- 여러 Options 가질 수 있음

Option(선택지)

- 여러 설문자에 의해 여러 답변에 선택될 수 있음

Choice(답변)

- 하나의 선택지만 고를 수 있음

User

- 문항 당 하나의 답변을 골라 여러 답변을 가질 수 있음

API 문서

유저 API

엔드포인트

유저 조회

요청

- **메소드:** `GET`
- **엔드포인트:** `/api/users`

응답

- **상태 코드:** `200 OK`
- **본문:**

```
{
  "data": [
    {
      "id": 1,
      "username": "user1"
    },
    // ... (다른 사용자들)
  ]
}
```

유저 등록

내부적으로 중복검증한다.

요청

- **메소드:** `POST`
- **엔드포인트:** `/api/users`
- **본문:**

```
{
  "username": "새로운사용자",
  "password": "비밀번호"
}
```

응답

- **상태 코드:** `200 OK`
- **본문:**

```
{
  "id": 1
}
```

유저 업데이트

요청

- **메소드:** `PATCH`
- **엔드포인트:** `/api/users/{id}`
  - `{id}`를 업데이트할 사용자의 ID로 교체합니다.
- **본문:**

```
{
  "username": "수정된사용자"
}
```

응답

- **상태 코드:** `200 OK`
- **본문:**

```
{
  "id": 1,
  "username": "수정된사용자"
}
```

유저 삭제

요청

- **메소드:** `DELETE`
- **엔드포인트:** `/api/users/{id}`
  - `{id}`를 삭제할 사용자의 ID로 교체합니다.

응답

- **상태 코드:** `200 OK`
- **본문:**

```
"삭제되었습니다."
```

유저 총 점수 조회

유저가 고른 답변의 점수 총합을 조회

요청

- **메소드:** `GET`
- **엔드포인트:** `/api/users/score/{id}`

응답

- **상태 코드:** `200 OK`

- **본문:**

```
100
```

요청 및 응답 모델

CreateUserRequest

```
{
  "username": "새로운사용자",
  "password": "비밀번호"
}
```

CreateUserResponse

```
{
  "id": 1
}
```

UpdateUserRequest

```
{
  "username": "수정된사용자"
}
```

UpdateUserResponse

```
{
  "id": 1,
  "username": "수정된사용자"
}
```

Result

```
{
  "data": [
    {
      "id": 1,
      "username": "user1"
    },
    // ... (다른 사용자들)
  ]
}
```

UserDto

```
{
  "id": 1,
  "username": "user1"
}
```

설문지 API

엔드포인트

설문 조회

요청

- **메소드:** `GET`
- **엔드포인트:** `/api/surveys`

응답

- **상태 코드:** `200 OK`
- **본문:**

```
[
  {
    "id": 1,
    "title": "첫 번째 설문",
```

```
    "description": "이 설문은 무엇에 관한 것인가요?",
    "questions": [
      {
        "id": 1,
        "text": "첫 번째 질문",
        "options": [
          // ... (다른 선택지들)
        ]
      },
      // ... (다른 질문들)
    ]
  },
  // ... (다른 설문들)
]
```

설문 저장

요청

- **메소드:** `POST`
- **엔드포인트:** `/api/surveys`
- **본문:**

```
{
  "title": "새로운 설문",
  "description": "이 설문은 무엇에 관한 것인가요?"
}
```

응답

- **상태 코드:** `200 OK`
- **본문:**

```
"저장되었습니다."
```

설문 업데이트

요청

- **메소드:** `PATCH`
- **엔드포인트:** `/api/surveys/{id}`
  - `{id}`를 업데이트할 설문의 ID로 교체합니다.
- **본문:**

```
{
  "title": "수정된 설문",
  "description": "이 설문은 무엇에 관한 것인가요?"
}
```

응답

- **상태 코드:** `200 OK`
- **본문:**

```
"수정되었습니다."
```

설문 삭제

요청

- **메소드:** `DELETE`
- **엔드포인트:** `/api/surveys/{id}`
  - `{id}`를 삭제할 설문의 ID로 교체합니다.

응답

- **상태 코드:** `200 OK`
- **본문:**

```
"삭제되었습니다."
```

요청 및 응답 모델

CreateSurveyRequest

```
{
  "title": "새로운 설문",
  "description": "이 설문은 무엇에 관한 것인가요?"
}
```

SurveyDto

```
{
  "id": 1,
  "title": "첫 번째 설문",
  "description": "이 설문은 무엇에 관한 것인가요?",
  "questions": [
    {
      "id": 1,
      "text": "첫 번째 질문",
      "options": [
        // ... (다른 선택지들)
      ]
    },
    // ... (다른 질문들)
  ]
}
```

QuestionDto

```
{
  "id": 1,
  "text": "첫 번째 질문",
  "options": [
    // ... (다른 선택지들)
  ]
}
```

오류 응답

- **상태 코드:** `400 Bad Request`
  - 잘못된 요청 형식 또는 필수 필드 누락.
- **상태 코드:** `404 Not Found`

    - 지정된 ID의 설문을 찾을 수 없음.
- **상태 코드:** `500 Internal Server Error`
    - 예상치 못한 오류 발생.

의존성

- **SurveyService:** 설문을 관리하는 서비스.

선택지 API

엔드포인트

선택지 조회

요청

- **메소드:** `GET`
- **엔드포인트:** `/api/options`

응답

- **상태 코드:** `200 OK`
- **본문:**

```
[
  {
    "id": 1,
    "optionText": "옵션1",
    "score": 10,
    "questionId": 123
  },
  // ... (다른 선택지들)
]
```

선택지 저장

요청

- **메소드:** `POST`
- **엔드포인트:** `/api/options`
- **본문:**

```
{
  "optionText": "옵션1",
  "score": 10,
  "questionId": 123
}
```

응답

- **상태 코드:** `200 OK`
- **본문:**

```
"저장되었습니다.
```

선택지 업데이트

요청

- **메소드:** `PATCH`
- **엔드포인트:** `/api/options/{id}`
  - `{id}`를 업데이트할 선택지의 ID로 교체합니다.
- **본문:**

```
{
  "optionText": "옵션1 수정",
  "score": 15,
  "questionId": 456
}
```

응답

- **상태 코드:** `200 OK`
- **본문:**

```
"수정되었습니다."
```

선택지 삭제

요청

- **메소드:** `DELETE`
- **엔드포인트:** `/api/options/{id}`
  - `{id}`를 삭제할 선택지의 ID로 교체합니다.

응답

- **상태 코드:** `200 OK`
- **본문:**

```
"삭제되었습니다."
```

요청 및 응답 모델

CreateOptionRequest

```
{
  "optionText": "옵션1",
  "score": 10,
  "questionId": 123
}
```

OptionDto

```
{
  "id": 1,
  "optionText": "옵션1",
```

```
  "score": 10,
  "questionId": 123
}
```

오류 응답

- **상태 코드:** `400 Bad Request`
  - 잘못된 요청 형식 또는 필수 필드 누락.
- **상태 코드:** `404 Not Found`
  - 지정된 ID의 선택지를 찾을 수 없음.
- **상태 코드:** `500 Internal Server Error`
  - 예상치 못한 오류 발생.

의존성

- **OptionService:** 선택지를 관리하는 서비스.
- **QuestionRepository:** 질문 정보를 검색하는 리포지토리.

답변 API

엔드포인트

답변 조회

요청

- **메소드:** `GET`
- **엔드포인트:** `/api/choices`

응답

- **상태 코드:** `200 OK`
- **본문:**

```
[
  {
    "choiceId": 1,
    "userName": "user1",
    "optionId": 123
  },
  // ... (다른 선택지들)
]
```

답변 저장

요청

- **메소드:** `POST`
- **엔드포인트:** `/api/choices`
- **본문:**

```
{
  "userName": "user1",
  "optionId": 123
}
```

응답

- **상태 코드:** `200 OK`
- **본문:**

```
"저장되었습니다."
```

답변 업데이트

요청

- **메소드:** `PATCH`
- **엔드포인트:** `/api/choices/{id}`
  - `{id}`를 업데이트할 선택지의 ID로 교체합니다.

- **본문:**

```
{
  "userName": "user1",
  "optionId": 456
}
```

응답

- **상태 코드:** `200 OK`
- **본문:**

```
"수정되었습니다."
```

답변 삭제

요청

- **메소드:** `DELETE`
- **엔드포인트:** `/api/choices/{id}`
  - `{id}`를 삭제할 선택지의 ID로 교체합니다.

응답

- **상태 코드:** `200 OK`
- **본문:**

```
"삭제되었습니다."
```

요청 및 응답 모델

CreateChoiceRequest

```
{
  "userName": "user1",
```

```
  "optionId": 123
}
```

ChoiceDto

```
{
  "choiceId": 1,
  "userName": "user1",
  "optionId": 123
}
```

오류 응답

- **상태 코드:** `400 Bad Request`
  - 잘못된 요청 형식 또는 필수 필드 누락.
- **상태 코드:** `404 Not Found`
  - 지정된 ID의 선택지를 찾을 수 없음.
- **상태 코드:** `500 Internal Server Error`
  - 예상치 못한 오류 발생.

의존성

- **ChoiceService:** 선택지를 관리하는 서비스.
- **UserRepository:** 사용자 정보를 검색하는 리포지토리.
- **OptionRepository:** 옵션 정보를 검색하는 리포지토리.

최적화 및 개발시 고려한 점들

엔티티

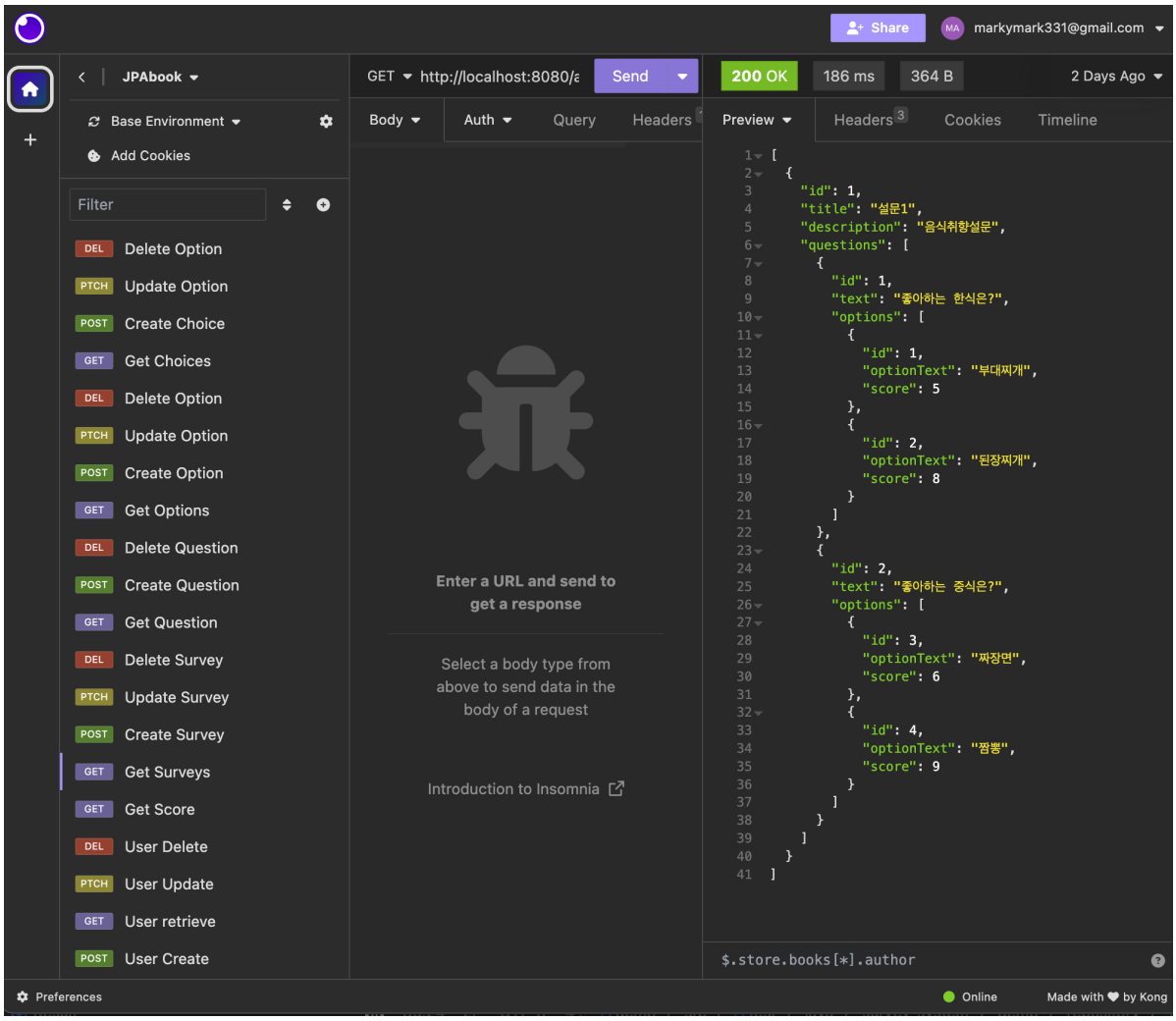
- 객체 중심의 테이블 모델링
- 모든 연관관계는 지연로딩으로 설정했다.
  - JPQL에서 N+1 문제 방지. 예상 불가능한 SQL방지..
  - .yml에 hibernate.default_batch_fetch_size=100으로 지연로딩 쿼리 성능 최적화
- 컬렉션은 필드에서 초기화했다.
  - null 문제로부터 안전하고, 엔티티가 영속화될 때 내장컬렉션으로 변경되는 걸 방지
- 양방향 관계 설정은 연관관계 편의 메소드를 사용한다.
- 연관관계의 주인은 모두 외래키가 있는 엔티티로 설정했다.

컨트롤러

- 생성자 주입
  - 순환참조방지. NullPointerException 방지. 필드 final선언(객체 불변성 보장)
- DTO
  - 엔티티 직접 접근과 노출을 방지
  - API 스펙 변경에 유연한 대처를 위해
- 커맨드와 쿼리를 분리

테스트

- Insomnia로 API 테스트



- InitDb로 더미데이터 주입
- JUnit5로 일부 유닛 테스트