2024-1 데이터베이스 (20471-03) TEAM PROJECT REPORT

이화여자대학교 백준/솔브닥 랭킹용 알고리즘 문제 추천 프로그램:

뀨업

Team 01 뀨엘

팀원:

박세은 (2248031)

정소은 (2271053)

정은채 (2271056)

전희원 (2276274)

최이경 (2276327)

I. 선택한 기관의 데이터베이스 필요성 설명 및 요구 분석

I-1. 프로그램 개요 및 데이터베이스 필요성

뀨업의 서비스 대상은 Problem Solving을 공부하는 이화여대 학생들이다. 백준 Online Judge와 Solved.ac이라는 온라인 알고리즘 문제풀이 사이트는 Problem Solving을 공부하는 전국의 학생들 사이에 압도적인 인기를 끌고 있다. 백준은 다양한 유형과 난이도를 갖춘 수많은 문제들을 제공하며, 개인 및 단체 (학교, 기업 등) 랭킹 시스템이 갖추고 있다. 이 랭킹 시스템이 개인 및 단체 간의 긍정적인 경쟁심을 불러 일으켜, 문제 풀이의 원동력이 되고, 백준 사이트 활성화에 기여했다고 할 수 있다.

그러나, 이화여대는 현재 백준과 Solved.ac에서 각각 단체 랭킹 126위, 219위에 지나지 않는다. 이는 신촌 인근 다른 대학교들과 비하여, 또 이화여대의 실제 실력에 비하여 터무니없이 낮은 수치이다. 대학교 익명 커뮤니티인 에브리타임을 통해 백준 랭킹에 대한 교내 학생들의 반응을 조사해보았다. 그 결과, 이화여대 백준 랭킹이 낮다는 것은 다소간에 인지하고 있으나, (1) 학생들이 알고리즘 문제를 푸는 횟수가 적고 (2) 어떻게 하면 단체 랭킹을 올릴 수 있는지 잘 알지 못한다는 것을 확인하였다.

따라서, 학교 대상으로 알고리즘 문화가 정착되지 못해 이화여대의 순위가 낮다는 문제를 해결하기 위하여, '랭킹 상승'을 원동력으로 이화여대생의 알고리즘 문제 풀이를 고취하는 프로그램을 개발하였다. 수백명의 이화여대 사용자, 수십개의 단체, 10,000여개의 문제 데이터를 저장해야 하고, 다수의 이화여대 사용자의 프로그램 사용을 지원해야 하므로 이를 관리하기 위한 데이터베이스가 필수적이다.

I-2. 요구사항 명세서

1. 사용자 정보 등록 : solved.ac에서 순위을 제공하는 API를 통해, 가입한 사용자의 푼 문제 수와 교내 랭크를 불러와서 고객 정보로 저장한다.

2. 사용자에 대해서 핸들, http://solved.ac/ 링크, 푼 문제 수, 교내 개인 랭크, 오늘의 추천 문제 찜하기 여부를 유지해야 한다. 사용자는 핸들로 식별된다.

3. 단체 순위 등록: solved.ac에서 순위을 제공하는 API를 통해, solved.ac에 등록된 모든 단체의 단체 이름, 등수, 푼 문제 수를 단체 정보로 저장한다.

4. 단체는 단체 이름으로 식별된다.

5. 순위 기능: 그룹 간 순위, 이화여대보다 한 순위 앞선 단체의 이름과 문제 차이 수, 이화여대 내 사용자 본인의 순위를 사용자에게 제공한다.

6. 알고리즘 문제: 서비스는 이화여대 학생 중 아직 아무도 풀지 않은 문제들을 저장하고

있다. 각 문제에 대해서 문제 번호, 문제 제목, 티어, 알고리즘 태그(분류), 푼 사람 수(이화여대 외부 사람들), 문제 링크 정보를 유지해야 한다. 문제는 문제 번호로 식별된다.

7. 오늘의 문제: 사용자에게 오늘의 추천 문제를 제공한다. 이화여대 학생이 안 푼 문제들을 푼 사람 수 (이화여대 외부 사람들)가 많은 순으로 정렬한다. 골드 5문제, 실버 5문제, 브론즈 5문제를 골라 전체 사용자에게 이 15문제를 추천한다.

8. 찜하기 기능: 각 사용자는 오늘의 추천 문제 중 한 번에 최대 한 개의 문제에 '찜하기'를 할 수 있다. 문제를 풀고 싶은 사용자가 문제를 클릭한 뒤 자신의 ID를 입력하면 해당 문제 옆에 찜한 사용자의 핸들이 뜬다. '찜하기'를 한 사용자가 24시간 안에 문제를 풀 지 못하면 해당 문제는 다시 오늘의 문제에 등록된다.

9. 알고리즘 별 안 푼 문제 보기 기능: 사용자에게 알고리즘 분류 별로 안 푼 문제들을 보여준다. 문제들을 다른 사용자 (이화여대 외부 사람들)가 많이 푼 순으로 정렬하여 보여준다.

10. 티어 별 안 푼 문제 보기 기능: 사용자에게 티어 별로 안 푼 문제들을 보여준다. 문제들을 다른 사용자 (이화여대 외부 사람들)가 많이 푼 순으로 정렬하여 보여준다.

- (IX. 비고 참고)

11. 같이 풀어요 (게시글) 기능: 사용자가 문제 번호, 게시글 제목 자신이 코드를 작성하고 있는 Github 링크, 자신의 핸들 (생략 가능), 비밀 번호를 게시글로 등록한다. 게시글은 자동 부여되는 게시글 ID로 식별된다.

11-1. 사용자가 게시글을 삭제하려 할 때, 게시글에 대한 비밀 번호를 검사하여 일치할 때만 삭제한다.

12. 다른 사용자들이 게시글을 통해 Github 링크에 접속하여, Github 상에서 협동하여 문제를 풀 수 있다.

13. 사용자는 복수 개의 게시글을 게시할 수 있고, 게시글 하나는 한 명의 사용자만 작성할 수 있다.

II. ER 다이어그램

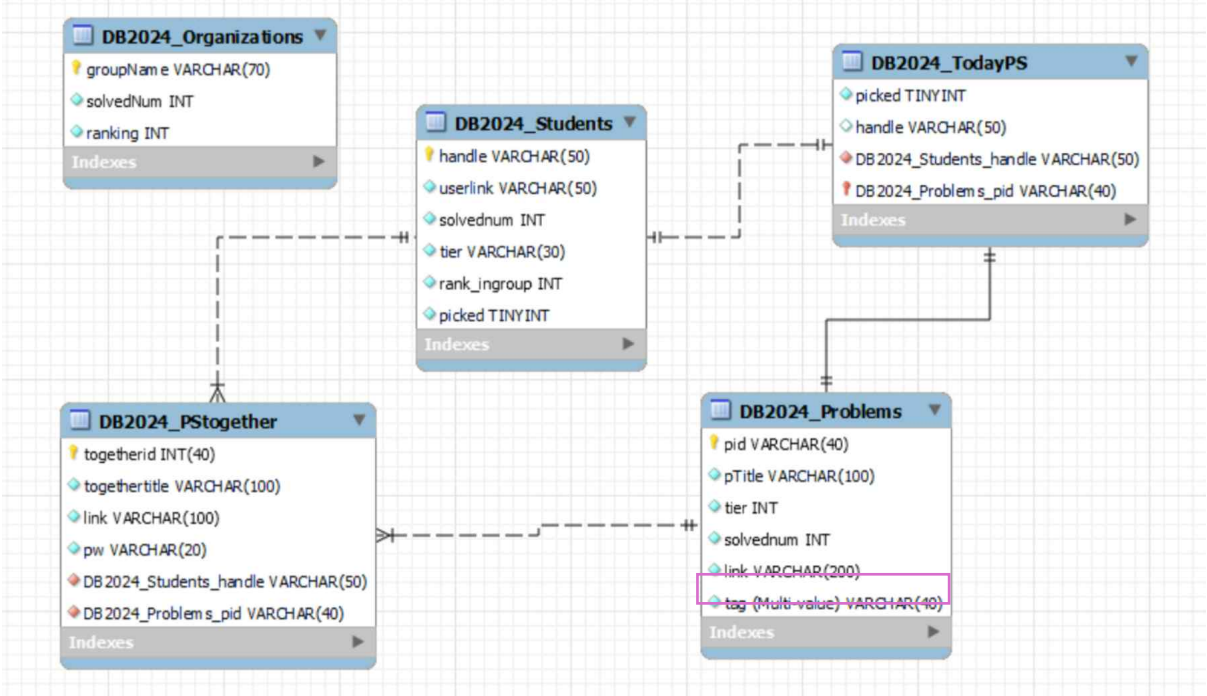


그림 . 다종값 속성을 처리하기 전의 ERD

II-1. 개체 타입 및 속성

DB2024_Students: 이화여자대학교 학생이며 Solved.ac에 가입된 학생 개체 타입

- handle VARCHAR(50): 기본키, 사용자 핸들
  - 핸들이란, 백준과 Solved.ac에서 사용자의 ID를 칭하는 말이다.
- userlink VARCHAR(50): 사용자 학생의 Solved.ac 계정 URL
- solvednum INT: 사용자 학생이 백준에서 푼 문제 개수
- tier varchar(30): 사용자(이화여대 학생)의 백준 티어
  - Tier(티어)란 백준에서 사용하는 일종의 레벨 시스템이다. 오름차순으로 Unrated, Bronze 5-1, Silver 5-1, Gold 5-1, Platinum 5-1, Ruby 5-1, Master의 단계가 있다.
- rank_ingroup INT: 사용자의 이화여대 내 개인 랭킹
- picked BOOLEAN(TINYINT): 사용자가 오늘의 추천 문제(TodayPS)를 한 문제 이상 '찜하기' 했는 지의 여부
  - 한 사람이 모든 문제를 '찜하기' 해 두어 다른 사용자의 풀이를 방해하는 것을 막기 위하여, 한 사람이 동시에 복수의 문제를 '찜하기'할 수 없도록 제한함

DB2024_Organizations: 이화여자대학교를 포함하여, 백준에 등록된 단체 개체 타입

- groupName VARCHAR(70): 기본키, 단체명
- ranking INT: 단체의 랭킹
- solvedNum INT: 단체에 속한 사용자들이 푼 문제 개수 합 (중복 없이 셈)

DB2024_Problems: 백준에서 제공하는 문제 개체 타입

- pid VARCHAR(40): 기본키, 문제 번호
  - 백준에서 이미 부여되어 있는 고유의 번호
- pTitle VARCHAR(100): 문제 제목
- tier INT: 문제 티어
  - Unrated, Bronze 5-1, Silver 5-1, Gold 5-1, Platinum 5-1, Ruby 5-1, Master로 구성되어 있지만, 이후 프로그램 구현의 편의를 위해 순서대로 0~36의 INT 값에 매칭시켜 저장
- solvednum INT: 백준 전체 사용자들에 의해 이 문제가 풀린 횟수
- link VARCHAR(200): 문제의 URL
- tag VARCHAR(20) (다중값 속성): 알고리즘의 카테고리를 백준과 Solved.dac에서 tag라고 칭함
  - 예) 문제 당 tag= sorting, backtracking 등 한 개 이상의 알고리즘 분류 태그를 가짐. 즉, 다중값 속성
  - 다중값 속성을 처리하기 위하여, 관계 데이터 모델로 사상할 때 DB2024_Problems 개체에서 DB2024_Algorthms를 분리할 예정

DB2024_TodayPS: 약한 개체, 오늘의 추천 문제 개체 타입

- pid VARCHAR(40): 기본키, 문제 번호
  - DB2024_Problems 개체의 pid 속성을 상속받아 기본키로 사용
- picked BOOLEAN: 추천 문제가 '찜하기' 되었는 지의 여부
- handle VARCHAR(50): 추천 문제 중 특정 문제를 '찜하기' 한 사용자의 핸들
  - DB2024_Students 개체의 handle 속성을 참조

DB2024_PStogether: 사용자들 간 문제 풀이를 공유할 수 있는 게시판에서, 게시글 개체 타입

- togetherid INT(40) AUTO_INCREMENT: 기본키, 게시글 번호
- togethertitle VARCHAR(100): 게시글 제목
- pid VARCHAR(40): 게시글로 질문하는 문제의 번호
  - DB2024_Problems 개체의 pid 속성을 참조
- handle VARCHAR(50): 게시자의 handle, 필수 입력 아님 (익명 가능)
  - null이 아닐 시, DB2024_Students 개체의 handle 속성을 참조

- link VARCHAR(100): 작성자의 github URL
  - ■ 게시글을 보고 도움을 주고자 하는 사용자가 github에 접속하여 협업 가능
- pw VARCHAR(20): 게시글 별 pw
  - ■ 게시글 등록 시 pw 등록, 후에 게시글 삭제 시 pw 입력하여 일치해야만 삭제

II-2. 관계

1. DB2024_PStogether와 DB2024_Students의 관계

- N:1 관계: 한 명의 Student가 여러 개의 게시글을 작성할 수 있다. 한 개의 게시글은 한 명에 의해서만 작성된다. 따라서 DB2024_PStogether가 N인 N:1 관계이다.
- 비식별자 관계: DB2024_Students의 handle 속성이 DB2024_PStogether의 일반 속성으로 포함된다.

2. DB2024_PStogether와 DB_Problems의 관계

- N:1의 관계: 한 개의 Problem은 여러 개의 게시글로 작성될 수 있다. 한 개의 게시글은 하나의 문제에 대해서만 작성된다. 따라서 DB2024_PStogether가 N인 N:1 관계이다.
- 비식별자 관계: DB2024_Problems의 pid 속성이 DB2024_PStogether의 일반 속성으로 포함된다.

3. DB2024_TodayPS와 DB2024_Students의 관계

- 1:1 관계: 한 명의 Student는 한 번에 한 문제만 '찜하기' 할 수 있다. 한 문제는 한 번에 한 명에 의해서만 '찜하기' 될 수 있다.
- 비식별자 관계: DB2024_Students의 handle 속성이 DB2024_TodayPS의 일반 속성으로 포함된다.

4. DB2024_TodayPS와 DB2024_Problems

- 1:1 관계: 한 개의 문제는 하루에 0번 혹은 1번만 추천문제가 된다. 한 개의 추천 문제는 하나의 문제에 대응된다.
- 식별자 관계: DB2024_Problems의 pid 속성이 DB2024_TodayPS의 기본키로 포함된다.

.

III. 데이터베이스 스키마 다이어그램

III-3. ERD → RDB 사상에서 다중 값 처리

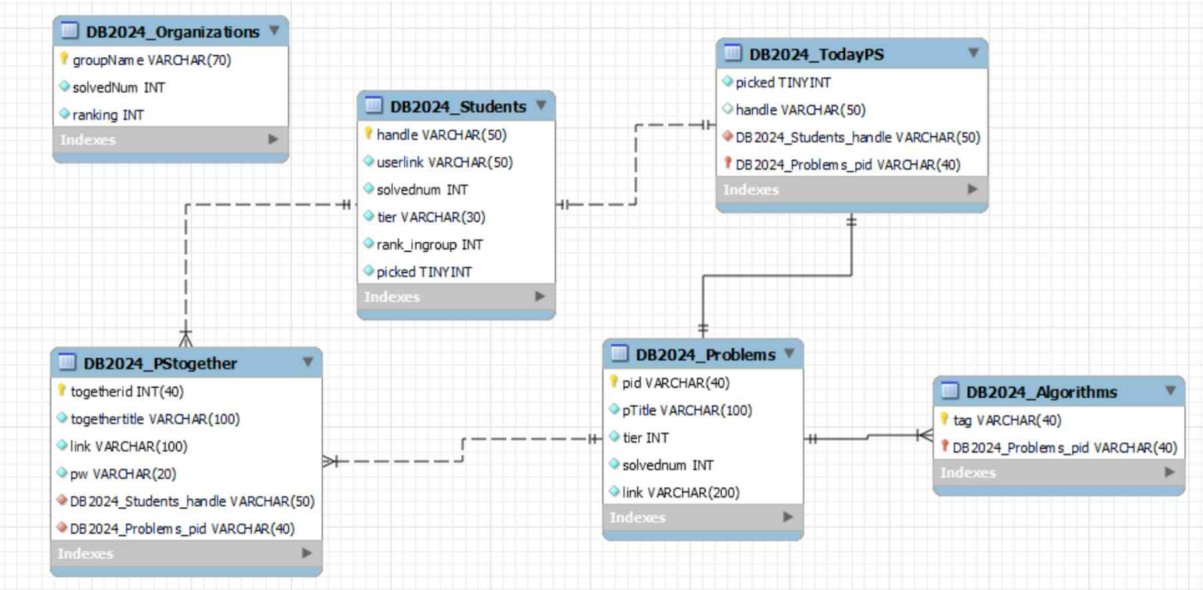


그림 . (참고) 관계 데이터 모델로 나타내기 위해 다중값 속성을 분리한 후의 ERD

DB2024_Problems에서 DB2024_Algorithms가 분리되었고, 이 둘은 관계를 생성한다.

N:1 관계: (문제 번호, tag)에 대해 한 개의 (문제 번호)가 대응된다. (문제 번호)에 대해 여러 개의 (문제 번호, tag)가 대응된다. 따라서 DB2024_Algorithms가 인 N:1 관계이다.

식별자 관계: DB2024_Problems의 pid 속성이 DB2024_Algorithms의 기본키로 포함된다. DB2024_Algorithms 개체는 다중값 속성인 tag를 처리하기 위해 분리되었는데, 분리될 때 원래 개체의 기본키 속성도 함께 분리되어야 분리된 개체 타입에서 개체 인스턴스 간 식별이 가능하다

III-2. 데이터베이스 스키마

DB2024_Students

Field	Type	Null	Key	Default	Extra
handle	varchar(50)	NO	PRI		
userlink	varchar(50)	NO			
solvednum	int	NO			
tier	varchar(30)	NO			
rank_ingroup	int	NO			

picked	tinyint(1)	NO		FALSE	DEFAULT_GENERATED

DB2024_Organizations

Field	Type	Null	Key	Default	Extra
groupName	varchar(70)	NO	PRI		
solvedNum	int	NO			
ranking	int	NO			

DB2024_Problems

Field	Type	Null	Key	Default	Extra
pid	varchar(40)	NO	PRI		
pTitle	varchar(100)	NO			
tier	int	NO			
solvednum	int	NO			
link	varchar(200)	NO			

DB2024_Algorithms

Field	Type	Null	Key	Default	Extra
pid	varchar(40)	NO	PRI (FOREIGN)		
tag	varchar(40)	NO	PRI		

DB2024_TodayPS

Field	Type	Null	Key	Default	Extra
pid	varchar(40)	NO	PRI		
picked	tinyint(1)	NO		0	
handle	varchar(50)	YES	FOREIGN		

DB2024_PSTogether

Field	Type	Null	Key	Default	Extra
togetherid	int	NO	PRI		auto_increment
togethertitle	varchar(100)	NO			
pid	varchar(40)	NO	FOREIGN		

link	varchar(100)	NO			
handle	varchar(50)	YES	FOREIGN		
pw	varchar(20)	NO			

IV. Java Program 설명

IV-1. Package 설명

- db2024Main : 뀨업의 메인페이지, 즉 이화여대의 백준 순위 기능과 오늘의 문제 기능의 구현 코드가 담긴 패키지
- db2024Problems : 이화여대가 안 푼 문제의 알고리즘별, 티어별 정렬 기능 구현 코드가 담긴 패키지
- db2024PStogether : 같이 풀어요 게시판 기능의 구현 코드가 담긴 패키지
- db2024Students : 이화여대 내 교내 랭킹 기능의 구현 코드가 담긴 패키지
- global : DBManager 관련 코드와 ApiResponse 관련 코드가 담긴 패키지

- controller : 프론트엔드와 API 소통할 때 사용하는 url과 특정 url에서 어떤 메서드를 호출해야 하는지 정의하는 코드가 담긴 패키지
- dto : 프론트엔드에게 데이터를 요청하거나 반환하는 형식을 정의하는 코드가 담긴 패키지
  - dto 패키지는 RequestDTO class와 ResponseDTO로 구성되어 있으며 RequestDTO는 프론트엔드로부터 메인페이지 기능 구현에 필요한 데이터를 받아와야 할 때 그 형식을 정의하는 DTO가 담긴 클래스이며 ResponseDTO는 백엔드에서 데이터를 반환할 때 그 형식을 정의하는 DTO가 담긴 클래스이다.

- service : 실질적으로 기능을 구현하는 코드가 담긴 패키지

IV-2. db2024Main

1) dto

(1) MainRequestDTO class

TodayPSDibInfoDTO class : 찜하기 기능 구현에 필요한, 프론트엔드로부터 받아와야 하는 정보인 pid(문제번호)와 handle(사용자 핸들)을 필드로 가지며 각 필드에 대해 getter와 setter가 정의되어 있다.

(2) MainResponseDTO class

GroupInfoDTO class : 그룹 정보 기능 구현에 필요한, 백엔드에서 반환해야 하는 정보인 ewha_ranking(이화여대 랭킹), rival_ranking(이화여대 전 순위 랭킹), rival_group_name(이화여대 전 순위 그룹 이름), solved_num_gap(푼 문제 수 차이)을 필드로 가지며 모든 필드를 파라미터로 한 생성자를 포함하고 있다

TodayPSDTO class : 오늘의 문제 기능 구현에 필요한, 백엔드에서 반환해야 하는 정보인 pid(문제번호), dib_handle(찜한 사용자 핸들), dib(찜하기 여부), p_title(문제 제목), p_tier(문제 티어)를 필드로 가지고 있으며 모든 필드를 파라미터로 한 생성자를 포함하고 있다

MainPageDTO : 메인페이지에 나타나야 하는 모든 데이터, 즉 앞선 GroupInfoDTO와 TodayPSDTO를 같이 담고 있으며 getter와 setter가 구현되어 있다

2) controller

(1) MainController class : 메인페이지 API 연결을 위한 클래스

- getMainPage() : https://localhost:8080/main 에 GET 명령을 내려, service 패키지의 getMainPage()를 실행시키고 반환 결과를 MainPageInfo에 담아 반환하는 함수
- enableTodayPSDib() : https://localhost:8080/dib/enable에 PUT 명령을 내리고 프론트엔드로부터 받은 TodayPSDibInfoDTO 형태의 사용자가 찜할 문제의 pid와 사용자의 handle을 service 패키지의 putTodayPSpicked()에 파라미터로 담아 실행시키고 반환 결과를 result에 담아 반환하는 함수

3) service

(1) MainPage class : 메인페이지 정보를 가져오는 코드가 담긴 클래스

- getGroupInfo() : DB2024_Organizations 테이블로부터 이화여대와 이화여대 전 순위 그룹의 백준 순위와 푼 문제 수를 가져온 뒤 두 그룹의 푼 문제 수 차이를 계산하고 GroupInfoDTO 형식으로 이화여대의 백준 순위, 전 순위 그룹의 백준 순위, 두 그룹의 문제 수 차이를 생성하고 반환하는 함수
- getTodayPS() : DB2024_TodayPS 테이블과 DB2024_Problems 테이블로부터 가져온 오늘의 문제의 문제 번호, 제목, 티어, 찜하기 여부, 찜한 핸들을 TodayPSDTO 형식으로 생성하고 반환하는 함수

- getMainPage() : getGroupInfo()와 getTodayPS()가 반환하는 각각의 DTO를 하나의 MainPageDTO로 합쳐서 반환하는 함수

(2) TodayPSpicked class : 오늘의 문제 찜하기 기능 구현 코드가 담긴 클래스

- putTodayPSpicked() : MainController의 enableTodayPSDib()를 통해 TodayPSDibInfoDTO 형식의 파라미터를 request 변수를 통해 받는다. request 내의 필드인 pid와 handle을 getter 형식으로 받아와 각각 pid, handle 변수에 저장하고 DB2024_TodayPS의 picked와 handle과 DB2024_Students의 picked를 업데이트해주고 "picked success"를 반환한다. 이때 DB2024_Students 에서 handle을 통해 찾은 튜플의 picked가 이미 True이거나 DB2024_TodayPS에서 pid를 통해 찾은 튜플의 picked가 True인 경우 update를 진행하지 않고 "picked rejected"를 반환한다.

IV-3. db2024Problems

1) dto

(1) ProblemResponseDTO

ProblemAlgoDTO class : 이화여대가 안 푼 문제를 알고리즘별로 정렬하는 기능 구현에 필요한, 백엔드에서 반환하는 정보인 pid(문제번호), p_title(문제 제목), link(문제 링크), solvednum(이화여대를 제외하고 몇명이 풀었는지), tier(문제 티어)와 모든 필드를 파라미터로 하는 생성자를 가지고 있다.

(2) ProblemTierDTO class : 이화여대가 안 푼 문제를 티어별로 정렬하는 기능 구현에 필요한, 백엔드에서 반환하는 정보인 pid(문제번호), p_title(문제 제목), link(문제 링크), solvednum(이화여대를 제외하고 몇명이 풀었는지)와 모든 필드를 파라미터로 하는 생성자를 가지고 있다.

2) controller

(1) ProblemController : 문제 정렬 기능 API 연결을 위한 클래스

- getProblemAlgo() : https://localhost:8080/problems/algo에 GET 명령을 내리고 프론트엔드로부터 받은 사용자가 입력한 알고리즘 태그(tag)를 url 내의

파라미터로 받아 getProblemsByTag() 에 파라미터로 전달하여 실행시키고 그 결과를 ProblemAlgoList에 저장하고 반환하는 함수

- getProblemTier() : https://localhost:8080/problems/tier 에 GET 명령을 내리고 프론트엔드로부터 받은 사용자가 입력한 티어(tier)를 url 내의 파라미터로 받아 getProblemsByTier()에 파라미터로 전달하여 실행시키고 그 결과를 ProblemTierList에 저장하고 반환하는 함수

3) service

(1) ProblemAlgo class : 문제를 알고리즘별로 정렬하는 기능 구현 코드가 담긴 클래스

- getProblemsByTag() : ProblemController의 getProblemAlgo()를 통해 사용자가 선택한 알고리즘 태그를 request 변수로 받는다. DB2024_Problems에서 해당 태그를 가진 모든 문제를 각각 ProblemAlgoDTO 형식으로 받고 모든 DTO들을 하나의 ArrayList에 삽입한 뒤 반환하는 함수
- getProblemsByTier() : ProblemController의 getProblemTier()를 통해 사용자가 선택한 티어를 tier 변수로 받는다. DB2024_Problems에서 해당 티어인 모든 문제를 각각 ProblemTierDTO 형식으로 받고 모든 DTO들을 하나의 ArrayList에 삽입한 뒤 반환하는 함수

IV-4. dbPStogether

1) dto

(1) PSTogetherRequestDTO

(1 – 1) PSTogetherSaveDTO class : 함께 풀어요 게시글 작성 기능에 필요한, 프론트엔드로부터 받아와야 하는 정보인 togethertitle(게시글 제목), handle(게시자 핸들), link(깃허브 링크), pw(게시글 비밀번호), pid(문제 번호)를 필드로 가지고 있으며 각각의 필드에 대해 getter를 가지고 있다.

(1 – 2) PSTogetherDeleteDTO class : 함께 풀어요 게시글 삭제 기능에 필요한, 프론트엔드로부터 받아와야 하는 정보인 togetherid(게시글 id), pw(비밀번호)를 필드로 가지고 있으며 각각의 필드에 대해 getter를 가지고 있다.

(2) PSTogetherResponseDTO

(2 – 1) PSTogetherDetailDTO class : 함께 풀어요 게시글 상세 조회 기능에 필요한, 백엔

드에서 반환해야 하는 정보인 togetherid(게시글 id), pid(문제번호), togethertitle(게시글 제목), handle(게시자 핸들), link(깃허브 링크), pw(비밀번호)를 필드로 가지고 있으며 모든 필드를 파라미터로 하는 생성자가 있다.

(2 – 2)PSTogetherPreviewDTO class : 함께 풀어요 게시글 프리뷰 조회 기능에 필요한, 백엔드에서 반환해야 하는 정보인 정보인 togetherid(게시글 id), pid(문제번호), togethertitle(게시글 제목)를 필드로 가지고 있으며 모든 필드를 파라미터로 하는 생성자가 있다.

2) controller

(1) PSTogetherController class : 함께 풀어요 게시판 API 연결을 위한 클래스

- PSTogetherSave() : https://localhost:8080/pstogether에 POST 명령을 내리고 프론트엔드로부터 PSTogetherSaveDTO 형식의 생성할 게시글 데이터를 request로 받고 psTogetherSave()에 파라미터로 전달하여 실행시키고 그 결과를 반환하는 함수
- getPSTogetherDetail() : https://localhost:8080/pstogether에 GET 명령을 내리고 프론트엔드로부터 상세 조회할 게시글의 id를 url 내의 파라미터로 전달받은 뒤 psTogetherGetDetail()에 파라미터로 전달하여 실행시키고 그 결과를 반환하는 함수
- getPSTogetherPreview() : https://localhost:8080/pstogether/previews 에 GET 명령을 내리고 getPSTogetherPreview()를 실행시킨 뒤 그 결과를 반환하는 함수
- getPSTogetherSearch() : https://localhost:8080/pstogether/search에 GET 명령을 내리고 프론트엔드로부터 url 내에 pid(문제번호)를 파라미터로 전달받은 뒤에 pid를 getPSTogetherSearch에 파라미터로 전달하여 실행시키고 그 결과를 반환하는 함수
- deletePSTogether() : https://localhost:8080/pstogether/delete에 DELETE 명령을 내리고 프론트엔드로부터 PSTogetherDeleteDTO 형식의 데이터를 전달받고 PSTogethercheckPW에 파라미터로 전달한 뒤 실행시키고 그 결과가 TRUE일 때만 PSTogetherDelete() 를 실행시키고 그 결과를 반환하는 함수

3) service

(1) PSTogetherSave class: 함께 풀어요 게시글 생성 기능 구현 코드가 담긴 클래스

- PSTogetherSave() : 게시글 생성에 필요한 데이터가 담긴 PSTogetherSaveDTO 형식의 데이터를 PSTogetherController를 통해 request 변수로 받는다. request로부터 pid(문제번호), link(깃허브 링크), handle(게시자 핸들), pw(비밀번호)를 추출하고 DB2024_PStogether에 insert한다.
- PSTogetherDetailDTO 생성자를 통해 DTO를 새로 생성하고 반환하는 함수

(2) PSTogetherGetDetail class : 게시글 상세 조회 기능 구현 코드가 담긴 클래스

- PSTogetherGetDetail() : 게시물 id를 PSTogetherController를 통해 전달받는다. DB2024_PStogether로부터 전달받은 게시물 id를 통해 해당 튜플을 찾는다. 튜플에서 togetherid(게시글 id), pid(문제번호), title(게시글 제목), handle(게시자 핸들), link(깃허브 링크), pw(비밀번호)를 추출하여 PSTogetherDetailDTO의 생성자에 파라미터로 삽입한 뒤 DTO를 생성 및 반환하는 함수

(3) PSTogetherGetPreview class : 게시글 프리뷰 조회 기능 구현 코드가 담긴 클래스

- PSTogehterGetPreview() : DB2024_PStogether로부터 모든 튜플을 select하고 각각의 튜플에서 togetherid(게시글 id), pid(문제번호), handle(게시자 핸들), togethertitle(게시글 제목)을 추출한 뒤 PSTogetherPreviewDTO 생성자를 통해 각각의 DTO를 생성하고 모든 PSTogetherDTO를 하나의 ArrayList에 삽입하여 반환하는 함수

(4) PSTogetherSearch class : 게시글 검색 기능 구현 코드가 담긴 클래스

- PSTogetherSearch() : pid(문제번호)를 PSTogetherController를 통해 전달받는다. DB2024_PStogether에서 전달받은 pid를 통해 해당하는 튜플을 모두 select한다. 각각의 튜플에서 togetherid(게시글 id), pid(문제번호), handle(게시자 핸들), togethertitle(게시글 제목)을 추출한 뒤 PSTogetherPreviewDTO 생성자를 통해 각각의 DTO를 생성하고 모든 PSTogetherDTO를 하나의 ArrayList에 삽입하여 반환하는 함수

(5) PSTogetherDelete class : 게시글 삭제 기능 구현 코드가 담긴 클래스

- PSTogethercheckPW() : 게시글 삭제를 위한 비밀번호 일치 여부 판단하는 함수. PSTogetherController로부터 게시글 삭제에 필요한 데이터를 PSTogetherDeleteDTO 형식으로 전달받고 delete 변수에 저장한다. delete로부터

삭제할 게시글 id(togetherid)와 사용자가 입력한 비밀번호(pw)를 추출하고 DB2024_PStogether로부터 togetherid를 통해 실제 비밀번호를 select하고 사용자 입력 비밀번호와 비교한다. 두 값이 일치하면 TRUE를, 일치하지 않으면 FALSE를 반환하는 함수

- PSTogetherDelete() : 게시글 삭제 함수. PSTogetherController를 통해 PSTogethercheckPW의 return값이 TRUE인 경우 실행되며 PSTogetherController를 통해 PSTogetherDeleteDTO를 전달받고 삭제할 게시글 id(togetherid)를 통해 DB2024_PStogether에서 해당 튜플을 delete하는 함수

IV-5. db2024Students

1) dto

(1) StudentRankResponseDTO class

StudentRankDTO class : 이화여대의 교내 랭크 기능 구현에 필요한, 백엔드에서 반환해야 하는 정보인 rank_ingroup(교내 랭크), handle(사용자 핸들), userlink(사용자 solved.ac 링크), tier(사용자 티어), solved_num(사용자가 푼 문제 수)를 필드로 가지고 있으며 모든 필드를 파라미터로 하는 생성자를 가지고 있다.

2) controller

(1) StudentRankController class : 교내 랭크 기능 API 연결을 위한 클래스

- getStudentRank() : https://localhost:8080/studentRank에 GET 명령을 내리고 getStudentsOrderedByRank()를 실행시킨 뒤 그 결과를 반환하는 함수

3) service

(1) StudentRank class : 이화여대 내 교내 랭크 기능 구현 코드가 담긴 클래스

- getStudentsOrderedByRank() : DB2024_Students의 모든 튜플을 select하고 각각의 튜플로부터 rank_ingroup(교내 랭크), handle(사용자 핸들), userlink(사용자 solved.ac 링크), tier(사용자 티어), solved_num(사용자가 푼 문제 수)를 추출한 뒤 생성자를 통해서 각 튜플에 대한 StudentRankDTO를 생성하고 생성한 DTO들을 ArrayList에 삽입하여 반환하는 함수

V. 프로그램 실행 방법

V-1. Frontend :프론트엔드 리액트 프로그램

1. Node.js 설치

- 설치 확인용 터미널 명령어 : npm -v
- 설치가 되지 않았다면 Node.js 공식 웹사이트로 이동하여 최신 LTS(Long-Term Support) 버전을 다운로드하여 설치한다.

2. 프로젝트 의존성 설치

- 프로젝트 디렉토리로 이동(frontend₩q_up-react) 한다.
- 의존성 설치용 터미널 명령어 : npm install

3. 실행

- 프로젝트 실행용 터미널 명령어 : npm start
- 프로젝트를 확인한다.
- URL : http://localhost:3000/
- 반응형 미구현으로 인하여 1920 X 1080 px 규격, 80% 를 권장

V-2. Backend :백엔드 스프링 프로그램

(Backend 이클립스 버전)

1. STS 플러그인 설치: 이클립스 마켓플레이스(Help -> Eclipse Marketplace)에서 'Spring Tools 4'를 검색하여 설치한다.

2. 설정: Project > Java Compiler > Annotation Processing > Enable annotation processing를 체크한다.

3. lombok 설치: lombok을 설치한 후 Project > Java Compiler > Annotation Processing > Factory Path에 External JARs로 추가한다.

4. 실행: QqyuUpApplication을 실행한다.

(Backend 인텔리제이 버전)

1. 실행: QqyuUpApplication을 실행한다.

VI. 17가지 요구사항 만족

(1) 5개 이상의 테이블을 가지고 있어야 하고 각 테이블들의 컬럼(Attribute)의 수를 합하면 20개 이상이어야 한다.

- 다음과 같이 6개의 테이블과 총 25개의 컬럼을 포함하고 있다,

DB2024_Students(handle, userlink, solvednum, tier, rank_ingroup, picked)

DB2024_Organizations(groupname, solvednum, ranking)

DB2024_Problems(pid, pTitle, tier, solvednum, link)

DB2024_Algorithms(pid, tag)

DB2024_TodayPS(pid, picked, handle)

DB2024_PStogether(togetherid, togethertitle, pid, link, handle, pw)

```
-- Students Table
CREATE TABLE DB2024_Students (
    handle VARCHAR(50) NOT NULL PRIMARY KEY,
    userlink VARCHAR(50) NOT NULL ,
    solvednum INT NOT NULL ,
    tier varchar(30) NOT NULL,
    rank_ingroup INT NOT NULL,
    picked boolean NOT NULL default (false)
);
-- Problems Table
CREATE TABLE DB2024_Problems (
    pid VARCHAR(40) NOT NULL PRIMARY KEY,
    pTitle VARCHAR(100) NOT NULL,
    tier int NOT NULL,
    solvednum INT NOT NULL ,
    link VARCHAR(200) NOT NULL
);
```

```
-- Organizations Table
CREATE TABLE DB2024_Organizations (
    groupName VARCHAR(70) NOT NULL PRIMARY KEY ,
    solvedNum INT NOT NULL ,
    ranking INT NOT NULL
);
-- Algorithms Table
CREATE TABLE DB2024_Algorithms (
    pid VARCHAR(40) NOT NULL,
    tag varchar(40) NOT NULL,
    PRIMARY KEY (pid, tag),
    FOREIGN KEY (pid) REFERENCES DB2024_Problems(pid)
);
```

```sql
-- TodayPS Table
CREATE TABLE DB2024_TodayPS (
    pid VARCHAR(40) NOT NULL PRIMARY KEY,
    picked boolean NOT NULL default false,
    handle VARCHAR(50),
    FOREIGN KEY (pid) REFERENCES DB2024_Problems(pid),
    FOREIGN KEY (handle) REFERENCES DB2024_Students(handle)
);
```

```sql
-- PStogether Table
CREATE TABLE DB2024_PStogether (
    togetherid INT(40)AUTO_INCREMENT NOT NULL PRIMARY KEY,
    togethertitle VARCHAR(100)  NOT NULL,
    pid VARCHAR(40) NOT NULL,
    link VARCHAR(100) NOT NULL ,
    handle VARCHAR(50) ,
    pw VARCHAR(20) NOT NULL,
    FOREIGN KEY (pid) REFERENCES DB2024_Problems(pid),
    FOREIGN KEY (handle) REFERENCES DB2024_Students(handle)
);
```

(2) 초기화를 위해 적어도 30개의 레코드(투플)를 가지고 있어야 한다. (모든 테이블들의 레코드 수의 총합이 30개 이상)

- 다음과 같이 DB2024_Students에 375개, DB2024_Organizations에 200개, DB2024_Problems에 26522개, DB2024_Algorithms에 58088개, DB2024_TodayPS에 15개의 투플을 가지고 있다. (DB2024_PStogether는 초기 0개)

✓	24	17:52:24	INSERT INTO DB2024_Students(handle, userlink, solvednum, tier, rank_ingroup) VALUES ('celina32...	375 row(s) affected Records: 375 Duplicates: 0 Warnings: 0
✓	25	17:52:24	INSERT INTO DB2024_Organizations (groupname, solvedNum, ranking) VALUES ('KAIST', 21942, 1)...	200 row(s) affected Records: 200 Duplicates: 0 Warnings: 0
✓	26	17:52:24		26522 row(s) affected Records: 26522 Duplicates: 0 Warnings: 0
✓	27	17:52:24		58088 row(s) affected Records: 58088 Duplicates: 0 Warnings: 0
✓	28	17:52:25	insert into DB2024_TodayPS (pid) select pid from DB2024_Problems where tier between 1 and 5 orde...	5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0
✓	29	17:52:25	insert into DB2024_TodayPS (pid) select pid from DB2024_Problems where tier between 6 and 10 ord...	5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0
✓	30	17:52:25	insert into DB2024_TodayPS (pid) select pid from DB2024_Problems where tier between 11 and 15 or...	5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0

(3) 기본 키(primary key), 외래 키(foreign key), not null 제약조건(not null constraints)를 포함해야한다.

- (1)에서 보인 바와 같이 DB2024_Students에 기본 키 handle, DB2024_Organizations에 기본 키 groupname, DB2024_Problems에 기본 키 pid, DB2024_Algorithms에 DB2024_Problems로부터의 외래 키 겸 기본 키 pid와 기본 키 tag, DB2024_TodayPS에 DB2024_Problems로부터의 외래 키 겸 기본 키 pid(중복된 문제를 추천하지 않음)와 DB2024_Students로부터의 외래 키 handle, DB2024_PStogether에 기본 키 togetherid, DB2024_Problems로부터의 외래 키 pid와 DB2024_Student로부터의 외래 키 handle이 있다. 또한 DB2024_TodayPS의 handle과 DB2024_PStogether의 handle을 제외한 모든 속성은 not null 제약조건을 포함하고 있다.

CONSTRAINT_CATALOG	CONSTRAINT_SCHEMA	CONSTRAINT_NAME	TABLE_SCHEMA	TABLE_NAME	CONSTRAINT_TYPE	ENFORCED
def	db2024team01	PRIMARY	db2024team01	db2024_algorithms	PRIMARY KEY	YES
def	db2024team01	db2024_algorithms_ibfk_1	db2024team01	db2024_algorithms	FOREIGN KEY	YES
def	db2024team01	PRIMARY	db2024team01	db2024_organizations	PRIMARY KEY	YES
def	db2024team01	PRIMARY	db2024team01	db2024_problems	PRIMARY KEY	YES
def	db2024team01	PRIMARY	db2024team01	db2024_pstogether	PRIMARY KEY	YES
def	db2024team01	db2024_pstogether_ibfk_1	db2024team01	db2024_pstogether	FOREIGN KEY	YES
def	db2024team01	db2024_pstogether_ibfk_2	db2024team01	db2024_pstogether	FOREIGN KEY	YES
def	db2024team01	PRIMARY	db2024team01	db2024_students	PRIMARY KEY	YES
def	db2024team01	PRIMARY	db2024team01	db2024_todayps	PRIMARY KEY	YES
def	db2024team01	db2024_todayps_ibfk_1	db2024team01	db2024_todayps	FOREIGN KEY	YES
def	db2024team01	db2024_todayps_ibfk_2	db2024team01	db2024_todayps	FOREIGN KEY	YES

(4) 적어도 2개의 뷰를 정의해야 한다.

- 8개의 태그(자료구조, 그래프 이론, 다이나믹 프로그래밍, 기하학, 수학, 구현, 그리디 알고리즘, 문자열)별 문제 목록을 보다 간단하고 빠르게 제공하기 위해 다음과 같이 DB2024_Problems 테이블과 DB2024_Algorithms 테이블을 사용하여 각 태그에 해당하는 문제들의 모음인 8개의 뷰 DB2024_VIEW_tag_data_structures, DB2024_VIEW_tag_graphs, DB2024_VIEW_tag_dp, DB2024_VIEW_tag_geometry, DB2024_VIEW_tag_math, DB2024_VIEW_tag_implementation, DB2024_VIEW_tag_greedy, DB2024_VIEW_tag_string을 정의하고 있다.

```
-- CREATE VIEW
CREATE VIEW DB2024_VIEW_tag_data_structures AS
SELECT
    p.pid,
    p.pTitle,
    p.link,
    p.solvednum,
    p.tier
FROM DB2024_Problems p NATURAL JOIN DB2024_Algorithms a
WHERE a.tag LIKE '자료구조';
```

```
CREATE VIEW DB2024_VIEW_tag_graphs AS
SELECT
    p.pid,
    p.pTitle,
    p.link,
    p.solvednum,
    p.tier
FROM DB2024_Problems p NATURAL JOIN DB2024_Algorithms a
WHERE a.tag LIKE '그래프 이론';
```

```
CREATE VIEW DB2024_VIEW_tag_dp AS
SELECT
    p.pid,
    p.pTitle,
    p.link,
    p.solvednum,
    p.tier
FROM DB2024_Problems p NATURAL JOIN DB2024_Algorithms a
WHERE a.tag LIKE '다이나믹 프로그래밍';
```

```
CREATE VIEW DB2024_VIEW_tag_geometry AS
SELECT
    p.pid,
    p.pTitle,
    p.link,
    p.solvednum,
    p.tier
FROM DB2024_Problems p NATURAL JOIN DB2024_Algorithms a
WHERE a.tag LIKE '기하학';
```

```sql
CREATE VIEW DB2024_VIEW_tag_math
AS SELECT pid, ptitle, link, solvednum, tier
   FROM DB2024_Problems
   WHERE pid IN (SELECT pid FROM DB2024_Algorithms WHERE tag = '수학');
```

```sql
CREATE VIEW DB2024_VIEW_tag_implementation
AS SELECT pid, ptitle, link, solvednum, tier
   FROM DB2024_Problems
   WHERE pid IN (SELECT pid FROM DB2024_Algorithms WHERE tag = '구현');
```

```sql
CREATE VIEW DB2024_VIEW_tag_greedy
AS SELECT pid, ptitle, link, solvednum, tier
   FROM DB2024_Problems
   WHERE pid IN (SELECT pid FROM DB2024_Algorithms WHERE tag = '그리디 알고리즘');
```

```sql
CREATE VIEW DB2024_VIEW_tag_string
AS SELECT pid, ptitle, link, solvednum, tier
   FROM DB2024_Problems
   WHERE pid IN (SELECT pid FROM DB2024_Algorithms WHERE tag = '문자열');
```

(5) 모든 테이블과 뷰의 이름들은 "DB2024_"라는 접두어를 가지고 있어야 한다.

- (1), (4)에서 보인 바와 같이 모든 테이블과 뷰의 이름은 **DB2024_**로 시작한다.

(6) 적어도 4개의인덱스를 정의해야 한다.

- 다음과 같이 4개의 인덱스 ranking_index (DB2024_Organizations의 ranking 속성을 사용하는 인덱스), pid_index (DB2024_PStogether의 pid 속성을 사용하는 인덱스), tag_index (DB2024_Algorithms의 tag 속성을 사용하는 인덱스), tier_index (DB2024_Problems의 tier, solvednum 속성을 사용하는 인덱스)를 정의하고 있다.

```sql
-- CREATE INDEX
CREATE INDEX ranking_index ON DB2024_Organizations (ranking);
CREATE INDEX pid_index ON DB2024_PStogether (pid);
CREATE INDEX tag_index ON DB2024_Algorithms (tag);
CREATE INDEX tier_index ON DB2024_Problems (tier, solvednum DESC);
```

(7) 인덱스를사용하는 쿼리들을 포함해야 한다.

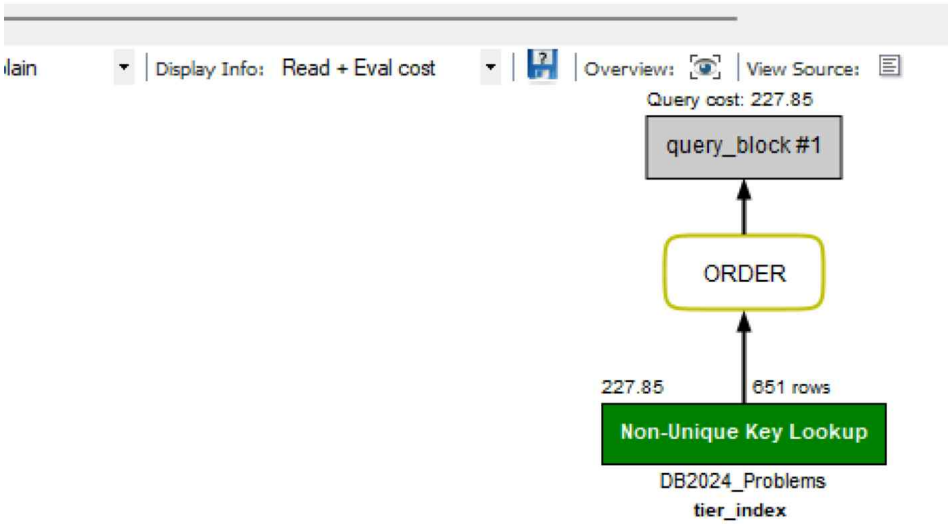
- 다음과 같은 쿼리들이 포함되어 인덱스를 사용하고 있다.

(7-1) tier_index: 티어별 문제 목록을 푼 사람이 많은 순으로 정렬하여 제공하는 getProblemsByTier(int tier) 함수에 포함된 쿼리

```
String query = "SELECT * FROM DB2024_Problems WHERE tier = ? ORDER BY solvednum DESC";
```

인덱스 사용예시)

```
SELECT * FROM DB2024_Problems WHERE tier = 7 ORDER BY solvednum DESC;
```

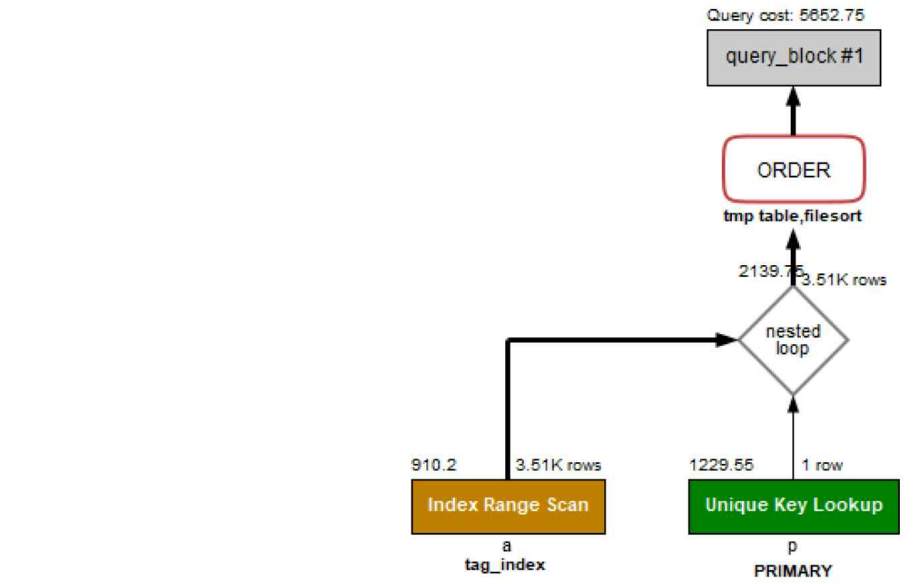


(7-2) solvednum_index: 태그(알고리즘)별 문제 목록을 푼 사람이 많은 순으로 정렬하여 제공하는 getProblemsByTag(String whichTag) 함수에 포함된 쿼리

```
String query = "SELECT * FROM DB2024_VIEW_tag_" + whichTag + " ORDER BY solvednum DESC";
```

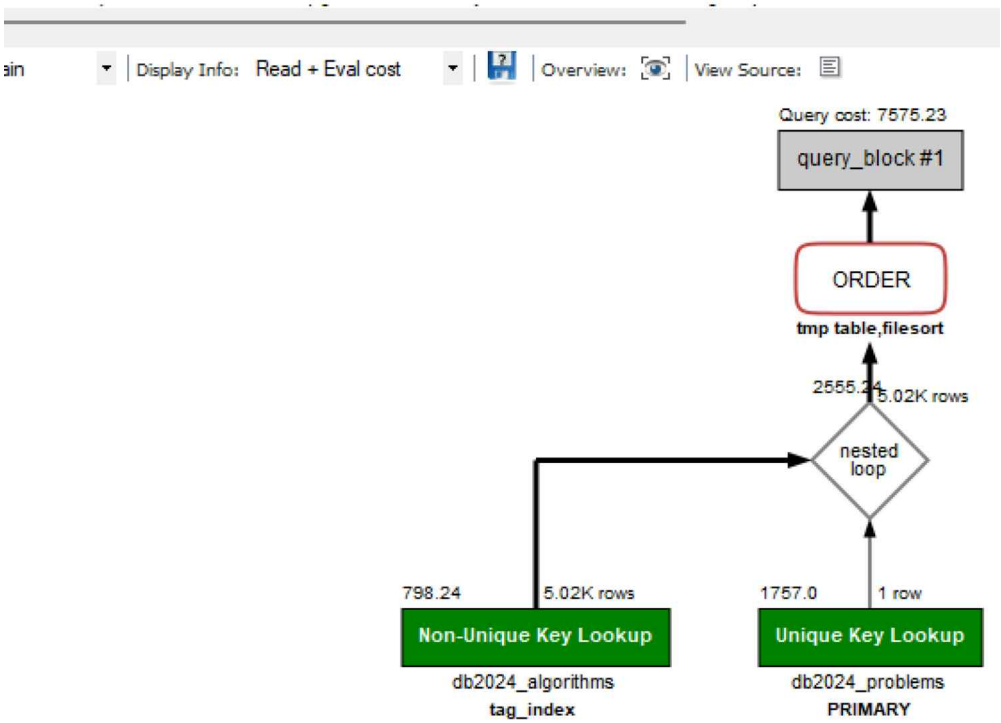
인덱스 사용예시 1)

```
SELECT * FROM DB2024_VIEW_tag_dp ORDER BY solvednum DESC;
```



인덱스 사용예시 2)

```
SELECT * FROM DB2024_VIEW_tag_math ORDER BY solvednum DESC;
```



(7-3) pid_index: '같이 풀어요' 게시판의 게시글을 문제 번호로 검색하는 PSTogetherSearch(int

whichPid) 함수에 포함된 쿼리

```
String query = "SELECT * FROM DB2024_PStogether WHERE pid="+whichPid;
```

(7-4) ranking_index: 이화여자대학교의 단체(그룹) 순위와 직전 순위 단체의 정보를 보여주는 getGroupInfo() 함수에 포함된 쿼리

```
PreparedStatement pstmt2 = conn.prepareStatement(
        "SELECT ranking, groupname, solvednum " +
                "FROM DB2024_Organizations " +
                "WHERE ranking = ((SELECT ranking FROM DB2024_Organizations WHERE groupname = ?) - 1)");

pstmt2.setString(1, "이화여자대학교");
```

인덱스 사용예시)

```
SELECT ranking, groupname, solvednum
FROM DB2024_Organizations
WHERE ranking = ((SELECT ranking FROM DB2024_Organizations WHERE groupname = "이화여자대학교") - 1);
```

(8) 뷰를 사용하는 쿼리를 포함해야 한다.

- 다음과 같은 쿼리가 포함되어 뷰를 사용하고 있다. 시용자는 다음 쿼리가 포함된 getProblemsByTag(String whichTag) 함수를 통해 자료구조, 그래프 이론, 다이나믹 프로그래밍, 기하학, 수학, 구현, 그리디 알고리즘, 문자열 중 하나의 태그를 선택해 그에 해당하는 문제들의 목록(각 문제의 문제 번호, 제목, 링크, 티어, 푼 사람 수를 제공)을 얻을 수 있다. 각 선택에서 whichTag에는 math, implementation, greedy, string, data_structures, graphs, dp, geometry가 들어가 DB2024_VIEW_tag_whichTag에 해당하는 뷰를 사용하게 된다.

```
String query = "SELECT * FROM DB2024_VIEW_tag_" + whichTag + " ORDER BY solvednum DESC";
```

(9) 트랜잭션(transaction)을 포함해야 한다.

- 오늘의 문제 '찜하기'는 한 사람 당 한 문제만 가능하다. 따라서 사용자가 자신의 핸들을 입력하고 '찜하기'를 누르면 DB2024_Students 테이블에서 handle이 사용자의 입력과 같은 투플을 찾아 picked를 true로 갱신하고 DB2024_TodayPS 테이블에서 사용자가 '찜하기'한 문제의 투플을 찾아 picked를 true로 갱신해야 한다. 이때 만약 두 갱신 작업 사이에 시스템 장애가 발생한다면 문제는 '찜하기' 되지 않았는데 사용자만 '찜하기'를 이미 한 상태가 되어 해당 사용자는 이후 그 어떤 문제도 '찜하기'를 할 수 없게 된다. 이런 현상을 방지하기 위해 다음과 같이 두 갱신 작업을 트랜잭션으로 묶어주었다.

putTodayPSpicked(int pid, String handle) 함수 일부:

```
PreparedStatement pstmt3 = conn.prepareStatement("update DB2024_TodayPS "
    + "set picked = true, handle = ? "
    + "where pid = ?");
PreparedStatement pstmt4 = conn.prepareStatement("update DB2024_Students "
    + "set picked = true "
    + "where handle = ?");
```

-해당 사용자 및 문제의 picked가 false인지 확인하는 코드 (생략)-

```
pstmt3.setString(1, handle);
pstmt3.setInt(2, pid);
pstmt4.setString(1, handle);

// 트랜잭션 시작
conn.setAutoCommit(false);

pstmt3.executeUpdate();
pstmt4.executeUpdate();

// 트랜잭션 끝
conn.commit();
conn.setAutoCommit(true);
```

(10) 중첩된 쿼리(nested query)들을 가지는 쿼리들을 포함해야 한다.

- 다음과 같은 중첩질의를 사용하는 쿼리들을 포함한다.

(10-1) 이화여자대학교의 직전 순위 단체에 대한 정보를 가져오는 쿼리: DB2024_Organizations 테이블에서 부속질의로 얻은 이화여자대학교의 순위 – 1 값과 ranking이 같은 투플을 반환한다.

```
PreparedStatement pstmt2 = conn.prepareStatement(
        "SELECT ranking, groupname, solvednum " +
                "FROM DB2024_Organizations " +
                "WHERE ranking = ((SELECT ranking FROM DB2024_Organizations WHERE groupname = ?) - 1)");

pstmt2.setString(1, "이화여자대학교");
```

(10-2) 태그별 문제 뷰를 정의하는 쿼리: 각각 DB2024_Algorithms 테이블에서 tag가 ('수학', '구현', '그리디 알고리즘', '문자열')인 문제의 번호들을 부속질의로 구하여 DB2024_Problems 테이블에서 해당하는 투플들만 반환하도록 하고 있다.

```
CREATE VIEW DB2024_VIEW_tag_math
AS SELECT pid, ptitle, link, solvednum, tier
   FROM DB2024_Problems
   WHERE pid IN (SELECT pid FROM DB2024_Algorithms WHERE tag = '수학');
```

```
CREATE VIEW DB2024_VIEW_tag_implementation
AS SELECT pid, ptitle, link, solvednum, tier
   FROM DB2024_Problems
   WHERE pid IN (SELECT pid FROM DB2024_Algorithms WHERE tag = '구현');
```

```
CREATE VIEW DB2024_VIEW_tag_greedy
AS SELECT pid, ptitle, link, solvednum, tier
   FROM DB2024_Problems
   WHERE pid IN (SELECT pid FROM DB2024_Algorithms WHERE tag = '그리디 알고리즘');
```

```
CREATE VIEW DB2024_VIEW_tag_string
AS SELECT pid, ptitle, link, solvednum, tier
   FROM DB2024_Problems
   WHERE pid IN (SELECT pid FROM DB2024_Algorithms WHERE tag = '문자열');
```

(11) 조인 쿼리(join query)들을 가지는 쿼리들은 포함해야 한다.

- 다음과 같은 조인을 사용하는 쿼리들을 포함한다.

(11-1) 오늘의 문제 목록을 출력하는 getTodayPS() 함수에 포함되는 쿼리: DB2024_Problems

테이블과 DB2024_TodayPS 테이블을 자연조인(pid 속성으로 조인됨)하여 문제 번호, 문제 제목, 찜하기 여부 등 오늘의 문제들의 정보를 반환한다.

```
ResultSet rs = stmt.executeQuery("select * "
        + "from DB2024_Problems natural join DB2024_TodayPS "
        + "order by tier");
```

(11-2) 태그별 문제 뷰를 정의하는 쿼리: 각각 DB2024_Problems 테이블과 DB2024_Algorithms 테이블을 자연조인(pid 속성으로 조인됨)하여 tag가 ('자료구조', '다이나믹 프로그래밍', '그래프 이론', '기하학')인 투플들만 반환하도록 하고 있다.

```
-- CREATE VIEW
CREATE VIEW DB2024_VIEW_tag_data_structures AS
SELECT
    p.pid,
    p.pTitle,
    p.link,
    p.solvednum,
    p.tier
FROM DB2024_Problems p NATURAL JOIN DB2024_Algorithms a
WHERE a.tag LIKE '자료구조';
```

```
CREATE VIEW DB2024_VIEW_tag_dp AS
SELECT
    p.pid,
    p.pTitle,
    p.link,
    p.solvednum,
    p.tier
FROM DB2024_Problems p NATURAL JOIN DB2024_Algorithms a
WHERE a.tag LIKE '다이나믹 프로그래밍';
```

```
CREATE VIEW DB2024_VIEW_tag_graphs AS
SELECT
    p.pid,
    p.pTitle,
    p.link,
    p.solvednum,
    p.tier
FROM DB2024_Problems p NATURAL JOIN DB2024_Algorithms a
WHERE a.tag LIKE '그래프 이론';
```

```
CREATE VIEW DB2024_VIEW_tag_geometry AS
SELECT
    p.pid,
    p.pTitle,
    p.link,
    p.solvednum,
    p.tier
FROM DB2024_Problems p NATURAL JOIN DB2024_Algorithms a
WHERE a.tag LIKE '기하학';
```

(12) 매개 변수를 가지면서 동적으로 만드는 쿼리를 포함해야 한다. 다시 말해, 사용자로부터 입력값을 받고 사용자가 입력한 값으로 쿼리를 생성한다.

- 다음과 같은 사용자의 입력값으로 완성하는 쿼리들을 포함한다.

(12-1) 티어별 문제 목록 제공 기능

DB2024_Problems 테이블에서 사용자가 고른 tier의 투플들을 반환하는 쿼리

```
String query = "SELECT * FROM DB2024_Problems WHERE tier = ? ORDER BY solvednum DESC";
```

(12-2) 태그별 문제 목록 제공 기능

사용자가 고른 태그에 해당하는 뷰에서 투플들을 반환하는 쿼리

```
String query = "SELECT * FROM DB2024_VIEW_tag_" + whichTag + " ORDER BY solvednum DESC";
```

(12-3) '같이 풀어요' 게시판 문제 번호 검색 기능

DB2024_PStogether 테이블에서 사용자가 입력한 pid(문제 번호)의 투플들을 반환하는 쿼리

```
String query = "SELECT * FROM DB2024_PStogether WHERE pid="+whichPid;
```

(12-4) 오늘의 문제 '찜하기' 기능

사용자가 원하는 오늘의 문제 pid와 사용자가 입력한 handle로 테이블을 갱신하는 쿼리

```
PreparedStatement pstmt3 = conn.prepareStatement("update DB2024_TodayPS "
    + "set picked = true, handle = ? "
    + "where pid = ?");
PreparedStatement pstmt4 = conn.prepareStatement("update DB2024_Students "
    + "set picked = true "
    + "where handle = ?");
```

(12-5) '같이 풀어요' 게시판 게시글 생성 기능

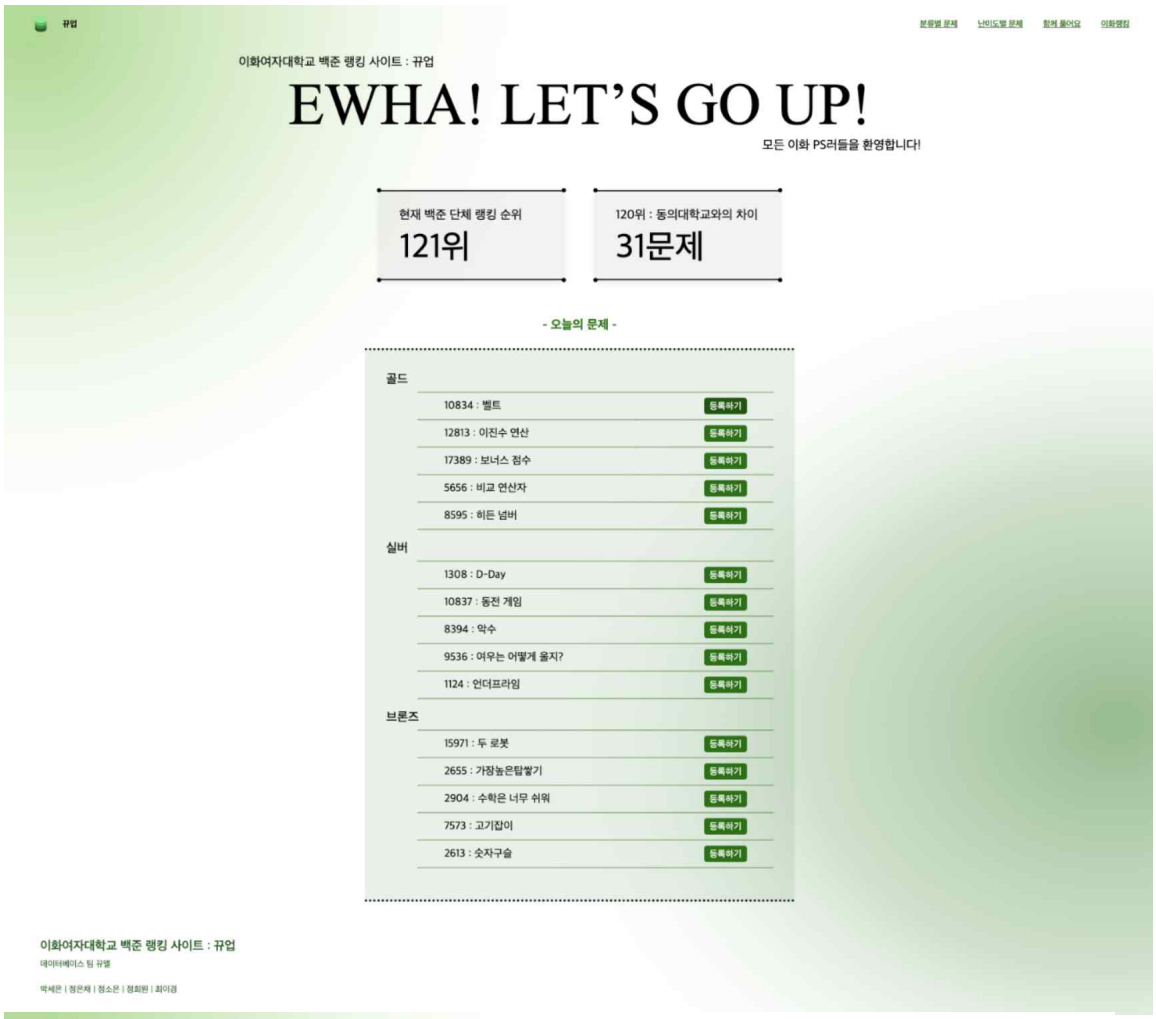
사용자가 입력한 대로 DB2024_PStogether 테이블에 투플을 삽입하는 쿼리

```
String sql = "INSERT INTO DB2024_PStogether (togethertitle, pid, link, handle, pw) VALUES (?, ?, ?, ?, ?)";
```

(13) 그래픽 또는 문자 기반의 사용자 인터페이스를 사용해야 한다.

사용하기 쉽고 사용하기에 도움이 되는 정보를 가지고 있는 메뉴를 제공해야 한다.

- 다음과 같이 리액트로 프론트엔드를 개발하여 사용자 친화적 인터페이스를 제공한다.



이화여자대학교 백준 랭킹 사이트 : 규업

데이터베이스 팀 규멜

박세은 | 정은채 | 정소은 | 정희원 | 최이경

등수	핸들	솔브닥 티어	푼 문제 개수	솔브닥 링크
1	celina324	Platinum 5	1046	https://solved.ac/profile/celina324
2	babeeboo2000	Platinum 2	1000	https://solved.ac/profile/babeeboo2000
3	pearl55	Platinum 4	796	https://solved.ac/profile/pearl55
4	iw0406	Platinum 4	773	https://solved.ac/profile/iw0406
5	kwakhj0205	Platinum 3	753	https://solved.ac/profile/kwakhj0205
6	meanjung	Platinum 4	621	https://solved.ac/profile/meanjung
7	emmajane	Gold 1	603	https://solved.ac/profile/emmajane
8	kha0318	Platinum 4	602	https://solved.ac/profile/kha0318
9	healing5746	Platinum 5	568	https://solved.ac/profile/healing5746
10	gaon0628	Platinum 5	556	https://solved.ac/profile/gaon0628
11	dk03006	Platinum 5	540	https://solved.ac/profile/dk03006
12	mummyee	Platinum 4	535	https://solved.ac/profile/mummyee
13	ye1111k	Platinum 5	524	https://solved.ac/profile/ye1111k
14	2000flora	Platinum 4	510	https://solved.ac/profile/2000flora
15	fungod12	Platinum 4	476	https://solved.ac/profile/fungod12
16	soultreeforgood	Gold 1	470	https://solved.ac/profile/soultreeforgood
17	zeomzzz	Gold 1	461	https://solved.ac/profile/zeomzzz
18	mkkw0228	Gold 1	456	https://solved.ac/profile/mkkw0228
19	pawken	Gold 1	449	https://solved.ac/profile/pawken

- 난이도별 벗들이 안 푼 문제 -

난이도별 분류
B5
B4
B3
B2
B1
S5
S4
S3
S2
S1
G5
G4
G3
G2
G1
P5

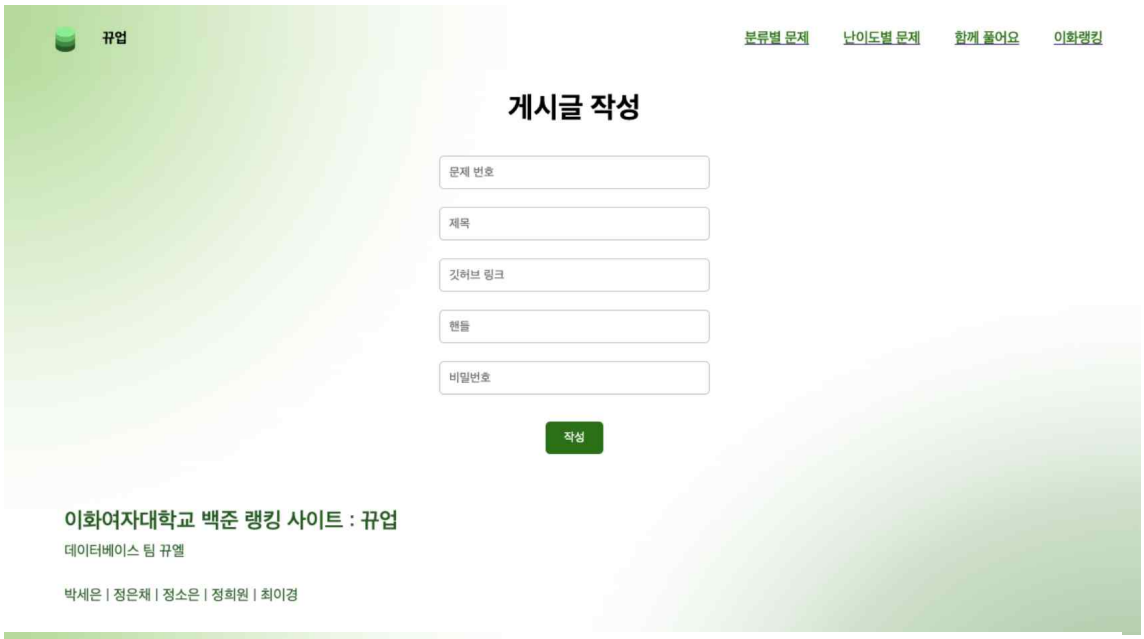
규업

분류별 문제 난이도별 문제 함께 풀어요 이화랭킹

- B2 -

번호	솔브닥 제목	푼 사람 수	푼 문제 링크
10040	투표	778	https://www.acmicpc.net/problem/10040
10182	AcidBase	185	https://www.acmicpc.net/problem/10182
10193	Word Swap	65	https://www.acmicpc.net/problem/10193
10202	Longest Subsequence	93	https://www.acmicpc.net/problem/10202
10203	Count Vowels	136	https://www.acmicpc.net/problem/10203
10230	Maze 1	205	https://www.acmicpc.net/problem/10230
10262	주사위 게임	495	https://www.acmicpc.net/problem/10262
10312	Lodê	144	https://www.acmicpc.net/problem/10312
10347	Reverse Rot	134	https://www.acmicpc.net/problem/10347
10410	Eligibility	105	https://www.acmicpc.net/problem/10410
10471	공간을 만들어 봅시다	265	https://www.acmicpc.net/problem/10471
10491	Quite a problem	226	https://www.acmicpc.net/problem/10491
10501	Ragged Right	88	https://www.acmicpc.net/problem/10501
10540	KLOPKA	292	https://www.acmicpc.net/problem/10540
10551	STROJOPIS	374	https://www.acmicpc.net/problem/10551
10643	FUNGHI	563	https://www.acmicpc.net/problem/10643
10675	Cow Routing	433	https://www.acmicpc.net/problem/10675

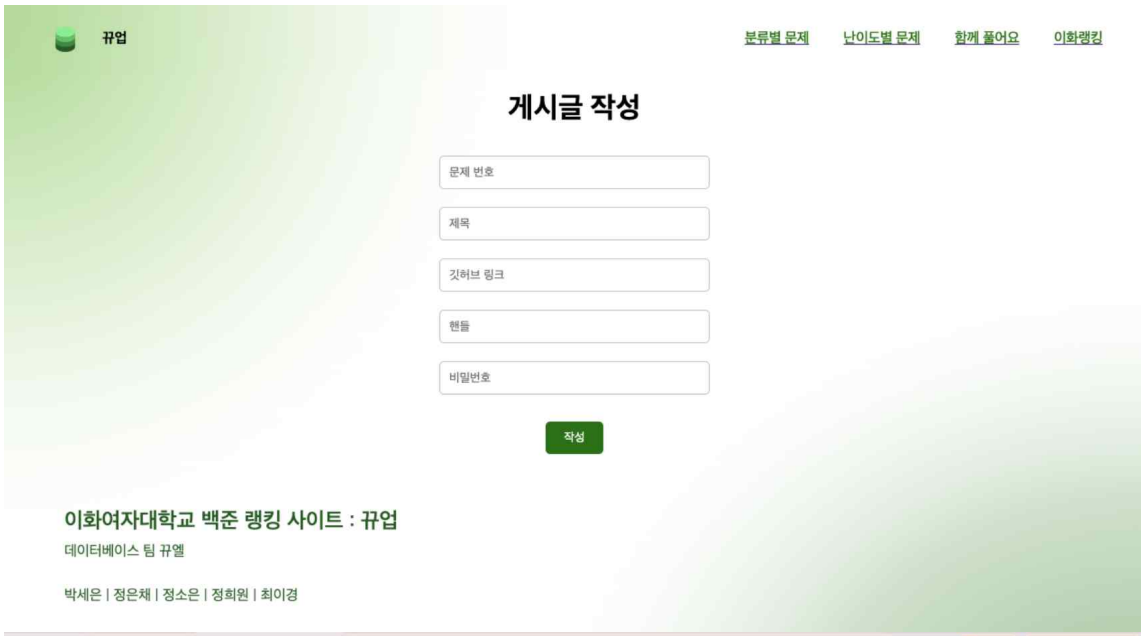
티어	번호	솔브닥 제목	푼 사람 수	푼 문제 링크
P3	1017	소수 쌍	2276	https://www.acmicpc.net/problem/1017
B2	10834	벨트	1967	https://www.acmicpc.net/problem/10834
S3	8394	악수	1955	https://www.acmicpc.net/problem/8394
S1	1124	언더프라임	1857	https://www.acmicpc.net/problem/1124
S3	10837	동전 게임	1767	https://www.acmicpc.net/problem/10837
S4	2980	도로와 신호등	1676	https://www.acmicpc.net/problem/2980
S1	15973	두 박스	1501	https://www.acmicpc.net/problem/15973
S1	1564	팩토리얼5	1334	https://www.acmicpc.net/problem/1564
P1	1067	이동	1326	https://www.acmicpc.net/problem/1067
B2	16769	Mixing Milk	1273	https://www.acmicpc.net/problem/16769
S5	15719	중복된 숫자	1261	https://www.acmicpc.net/problem/15719
P1	5615	아파트 임대	1222	https://www.acmicpc.net/problem/5615

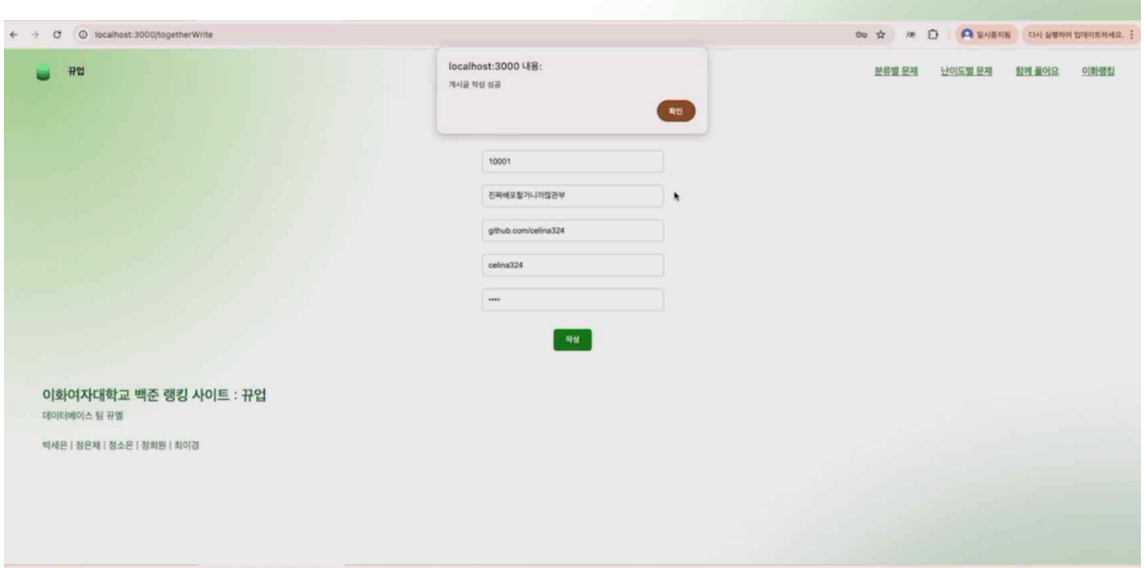


(14) 데이터베이스에 삽입(insert)을 하기 위한 인터페이스와 쿼리를 가지고 있어야 한다.

- 사용자는 '같이 풀어요' 게시판에 게시글을 올림으로써 DB2024_PStogether 테이블에 투플을 삽입할 수 있다. 이를 위해 다음과 같은 데이터베이스에 삽입(insert)하기 위한 쿼리와 인터페이스를 포함한다.

```
String sql = "INSERT INTO DB2024_PStogether (togethertitle, pid, link, handle, pw) VALUES (?, ?, ?, ?, ?)";
```


규업
분류별 문제
난이도별 문제
함께 풀어요
이화랭킹
게시글 작성
문제 번호
제목
깃허브 링크
핸들
비밀번호
작성
이화여자대학교 백준 랭킹 사이트 : 규업
데이터베이스 팀 규엘
박세은 | 정은채 | 정소은 | 정희원 | 최이경

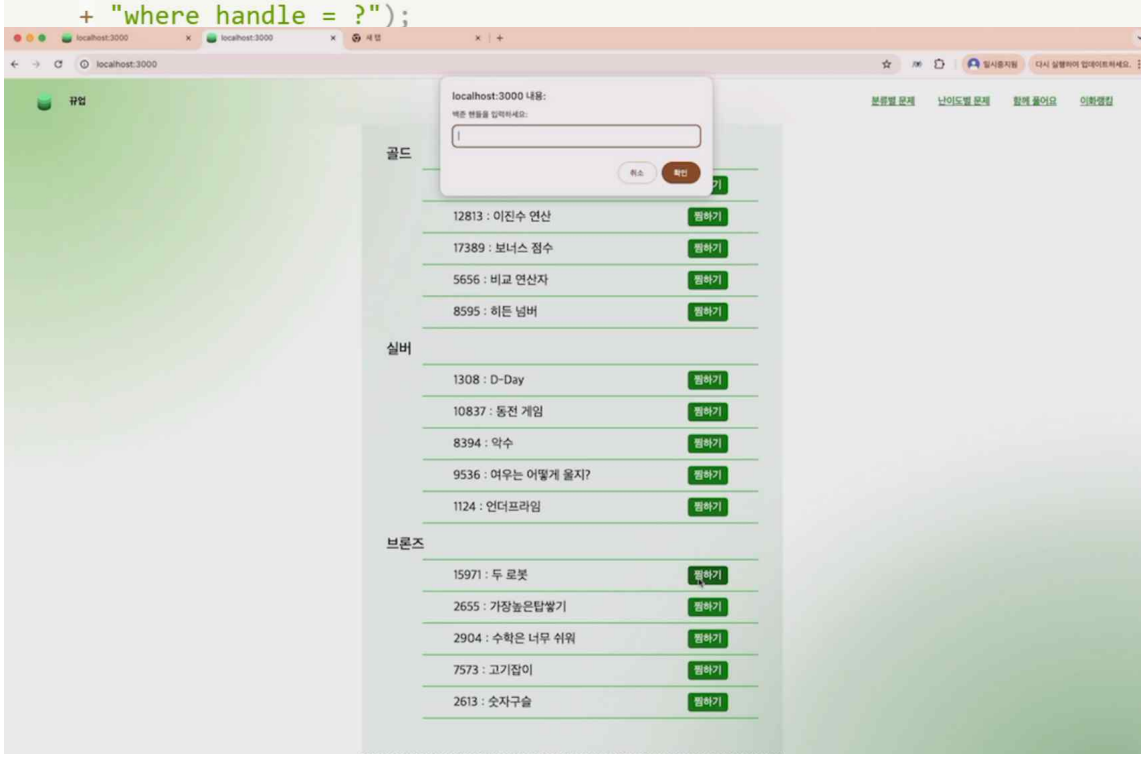

localhost:3000/togetherWrite
localhost:3000 내용:
게시글 작성 성공
확인
규업
분류별 문제
난이도별 문제
함께 풀어요
이화랭킹
10001
github.com/celina324
celina324
작성
이화여자대학교 백준 랭킹 사이트 : 규업
데이터베이스 팀 규엘
박세은 | 정은채 | 정소은 | 정희원 | 최이경


localhost:3000/together
규업
분류별 문제
난이도별 문제
함께 풀어요
이화랭킹
함께 풀어요
글쓰기
Search
문제 번호
게시글 제목
핸들
삭제
10009
제발
celina324
삭제
10001
햄스터
celina324
삭제
10009
세계정복
celina324
삭제
10001
진짜배포할거니까답관부
celina324
삭제
이화여자대학교 백준 랭킹 사이트 : 규업
데이터베이스 팀 규엘
박세은 | 정은채 | 정소은 | 정희원 | 최이경

(15) 데이터베이스에 갱신(update)을 하기 위한 인터페이스와 쿼리를 가지고 있어야 한다.

- 사용자가 오늘의 문제를 '찜하기' 하면 DB2024_TodayPS 테이블과 DB2024_Students 테이블의 투플들을 갱신해야 한다. 이를 위해 다음과 같은 데이터베이스에 갱신(update)하기 위한 쿼리와 인터페이스를 포함한다.

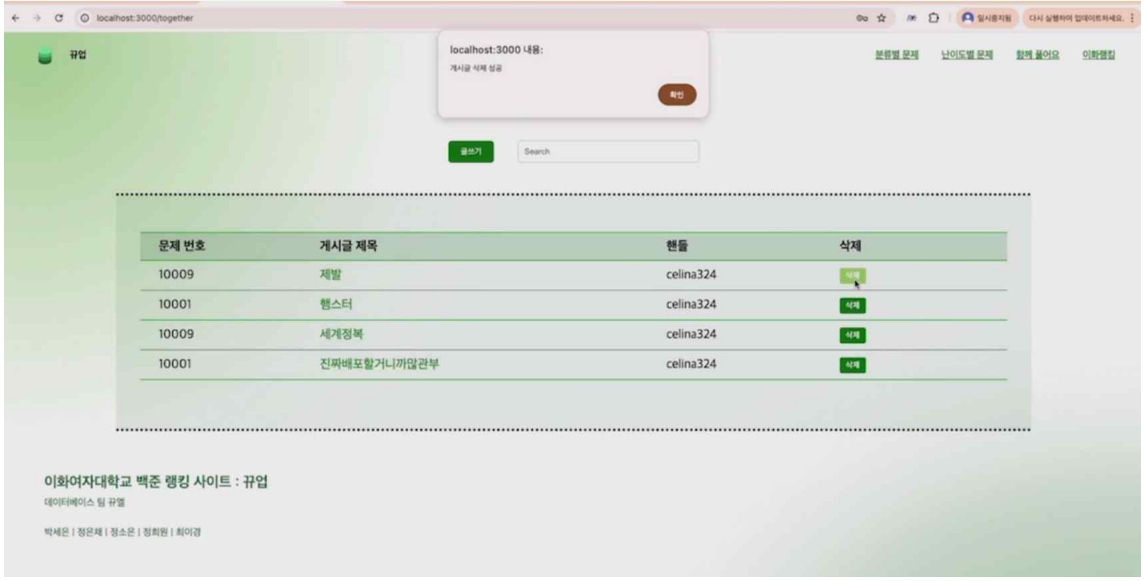
```
PreparedStatement pstmt3 = conn.prepareStatement("update DB2024_TodayPS "
    + "set picked = true, handle = ? "
    + "where pid = ?");
PreparedStatement pstmt4 = conn.prepareStatement("update DB2024_Students "
    + "set picked = true "
    + "where handle = ?");
```



(16) 데이터베이스에 삭제(delete)를 하기 위한 인터페이스와 쿼리를 가지고 있어야 한다.

- 사용자는 '같이 풀어요' 게시판의 게시글을 삭제할 수 있다. 이때 DB2024_PStogether 테이블에서 투플을 삭제(delete)해야 하므로 다음과 같은 쿼리와 인터페이스를 포함한다.

```
String query = "DELETE FROM DB2024_PStogether WHERE togetherid = ?";
```



(17) 데이터베이스에 검색(select)을 하기 위한 인터페이스와 쿼리를 가지고 있어야 한다.

- 사용자는 이화여자대학교의 순위, 앞 순위와의 격차, 오늘의 문제 목록, 태그별 문제 목록, 티어별 문제 목록, '같이 풀어요' 게시판 게시글 목록, 게시글 상세 조회, '같이 풀어요' 게시판 문제 번호로 검색한 게시글 목록 등을 제공받을 수 있으며 이를 위해 다음과 같은 검색(select) 쿼리들과 **(13)**에서 보인 바와 같은 인터페이스를 포함한다.

```
PreparedStatement pstmt2 = conn.prepareStatement(
        "SELECT ranking, groupname, solvednum " +
                "FROM DB2024_Organizations " +
                "WHERE ranking = ((SELECT ranking FROM DB2024_Organizations WHERE groupname = ?) - 1)");
```

```
pstmt2.setString(1, "이화여자대학교");
```

```
ResultSet rs = stmt.executeQuery("select * "
        + "from DB2024_Problems natural join DB2024_TodayPS "
        + "order by tier");
String query = "SELECT * FROM DB2024_VIEW_tag_" + whichTag + " ORDER BY solvednum DESC";
String query = "SELECT * FROM DB2024_Problems WHERE tier = ? ORDER BY solvednum DESC";
String query = "SELECT * FROM DB2024_PStogether";
String query = "SELECT * FROM DB2024_PStogether WHERE pid="+whichPid;
```

VII. 우리 팀의 강점

1. 데이터베이스 지식이 전무한 사용자가 부드럽게 사용할 수 있도록, 사용자 친화적인 UI를 구현했다.
2. 웹 크롤링으로 실제 데이터를 수집하였으므로, 실시간성 지원 등 조금의 보강을

가하면 사용자들이 실제로 프로그램을 사용하여 효용을 얻을 수 있다.

3. 총 튜플 개수 80,000여 개에 달하는 풍부한 데이터를 다룬다.
4. 다중값 속성을 가지는 개체를 관계데이터모델로 사상했다.
5. '같이 풀어요' 기능에 비밀번호 기능을 도입하여 아무나 게시글 삭제를 할 수 없도록 제한한다.

VIII. 팀원 역할 담당

박세은 (2248031): 게시글 작성 페이지 프론트엔드 구현 및 api 연동, 프론트엔드 페이지 수합, PPT 제작

정소은 (2271053): API 명세서 작성, 같이 풀어요 게시판 SQL 스크립트 작성, 백엔드 API 구현, 레포트 IV번 작성, PPT 제작, 최종 발표

정은채 (2271056): 프론트엔드 기본 틀, 메인 페이지 / 알고리즘 태그별 페이지 / 난이도별 페이지 / 랭킹 페이지 구조 구현 및 백엔드 api 연동, 영상 촬영, PPT 제작, 레포트 V번 작성, 중간 발표

전희원 (2276274): 데이터 수집 (백준 웹사이트 크롤링), sql스크립트 작성, 백엔드 코드 초안 작성, 레포트 VI번 작성, PPT 제작, 중간 발표

최이경 (2276327): 개발 일정 계획 수립, sql 스크립트 작성 및 주석 첨부, 백엔드 코드 초안 작성, 보고서 I, II, III번 작성 및 보고서 전체 관리, PPT 제작, 최종 발표

IX. 비고

티어 별 문제 보기 기능에서 Bronze 5에 해당하는 문제가 보이지 않는데, 그 이유는 Bronze 5에 해당하는 문제가 백준에 존재하지 않기 때문이다.

tier=unrated인 경우 UI에서 공란으로 표기하였다.