



지방상수도 수질 정보 제공 및 제보게시판 운영

GS ITM 3기 1조

팀장 유병진 - jeremy2695@gmail.com

팀원 김동현 - sadyour@gmail.com

천연수 - cheonyeonsu@gmail.com

황제희 - qlqj47@naver.com

양진웅 - jinwong936592@gmail.com

이정근 - junggnl@naver.com

01

프로젝트개요

1-1. 개발동기

1-2. 기획의도

01 프로젝트개요

1-1. 개발동기

1-2. 기획의도

개발동기

한국경제TV · 5일 전 · 네이버뉴스

더러운 수돗물 '칼칼'...인천 송도 '발각'

인천 송도국제도시 일대의 수돗물 수질이 악화되어 24시간 넘게 주민들이 불편을 겪고 있다. 26일 오후 2시 기준 연수구 송도 2·5동 일대에 공급되는 수돗물의 탁도가 최대 3.56NTU를 기록했다고 인천시 상수도사업...

인천 송도서 이틀째 음용 부적합 수돗물..급수차 지원 MBC · 4일 전 · 네이버

인천 송도 이틀째 수돗물 공급 제대로 안돼...불순물 섞여 나와 뉴스핌 · 4일

인천 송도서 24시간 넘게 '음용 부적합' 수돗물 칼칼 연합뉴스 PICK · 5일 전

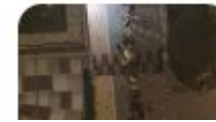
인천 송도 1만 가구 '음용 부적합' 수돗물 24시간 넘게 공급 조선일보 · 5일 전

관련뉴스

데일리한국 · 5일 전

여름 다가오자 유충 발견됐던 창원시 수질관리 '비상'

사진=창원시 수돗물과 수영장 등에서 갈따구 유충이 잇따라 발견돼 홍역을 치렀던 창원시가 올 여름에도 폭염이 예상되면서 수질관리에 비상이 걸렸다. 창원시는 수돗물 생산시설인 5개 정수장과 배수지 및 수도꼭지...



뉴스프리존 · 6일 전

창원특레시, 맑고 깨끗한 수돗물 생산 '만전'

유충)의 증식이 우려되고 있는 가운데 창원특레시가 정수장 정수 처리 공정 점검에 나섰다. 장금용 제1부사장은 24일 대산정수과에서 소형생물 대응을 위해 중점적으로 추진하고 있는 소형생물 대응 여과망 설치사업과...



경기신문 · 3주 전

남양주시, 수돗물 유충 발생 차단 시설 개선 중

미세여과망 설치 및 여과지 유충 유입방지 시설 등 주광덕 시장 "수돗물 안심하고 음용할 수 있도록 최선" 남양주시가 수돗물 유충(갈따구) 발생을 예방하기 위해 위생관리를 강화하고, 정수장 내 유충 발생 차단시설을 ...



창원시 깨끗하고 안전한 수돗물 생산 만전 기해 국제뉴스 · 3주 전

창원시, 여름철 깨끗하고 안전한 수돗물 생산 '총력' 메트로신문 · 3주 전

창원시, 여름철 수질 악화 대비 정수 시설을 집중점검 브릿지경제 · 3주 전

창원특레시, 깨끗하고 안전한 수돗물 생산 총력 뉴스프리존 · 3주 전



01 프로젝트개요

1-1. 개발동기

1-2. 기획의도

기획의도

공급되는 수돗물의 pH, 탁도, 잔류염소 농도의 등의 수질 적합 기준과 측정치에 대한 정보를 제공.
수질정보 제보게시판을 통해 이용자들의 정보교환을 용이하게 함.

(* 프로젝트 내 정보제공에 사용하는 API의 특성 상 지방의 특정 지역을 중심으로 정보를 제공함.)



우리집만 수질이
문제가 있는 건가?
나랑 같은 지역 사람들도
문제가 있는 건가?



현재 우리집으로 공급되
는 수돗물의 상태는 어느
정도지?

02

진행프로세스

2-1. 개발일정

2-2. 개발환경

2-3. 개인역할

02 진행프로세스

2-1. 개발일정

2-2. 개발환경

2-3. 개인역할

개발일정

06.19 ~ 06.20

프로젝트 제안 및 주제 선정



06.21 ~ 06.23

ERD 설계 / 기획 및 기능 관련 아이디어 정리



06.24 ~ 07.01

개발 및 TEST



07.01 ~ 07.02

검토 및 기능 개선



07.01 ~ 07.02

최종 TEST 및 발표자료 작성



07.03

프로젝트 발표



02 진행프로세스

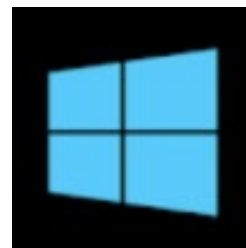
2-1. 개발일정

2-2. 개발환경

2-3. 개인역할

개발환경

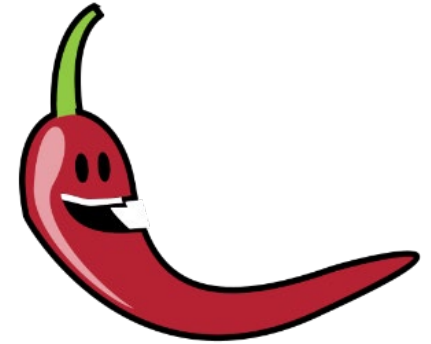
Back-End



Front-End



Tools



02 진행프로세스

2-1. 개발일정

2-2. 개발환경

2-3. 개인역할

팀장 유병진

- 프로젝트 총괄
- 제보게시판 페이지네이션 기능 구현
- 로그인/비로그인 상태를 구분하여 제보게시판의 각 기능에 대해 접근할 때 작동이 다르도록 구현
- 게시글 추천/추천 취소 기능 구현
- 회원가입시 이메일 인증 구현

팀원 이정근

- 제보게시판 및 댓글 생성/수정/삭제 및 조회 구현
- 각 제보게시글에 접근할 때를 기준으로 조회수를 산정하고, 게시글에 이에 대한 정보를 누적하면 표현하는 기능 구현
- 게시글에 이미지를 업로드하고, 이를 조회화면에서 확인할 수 있도록 기능 구현

팀원 천연수

- 회원가입 페이지 구현(유효성 검사)
- 로그인 페이지 구현
- 회원정보 수정 및 회원탈퇴 기능 구현
- 스프링 시큐리티 관련 Config 내용 작성

팀원 황제희

- ERD 설계 및 ERD Diagram 제작
- 마이 페이지 / 작성자 글 목록, 댓글 목록 확인 페이지 구현
- 프로젝트 내 전반적인 CSS 담당

팀원 양진웅

- 공공데이터 API를 활용하여
- Main 페이지에서 사용자가 선택한 지역 및 날짜에 대한 수질정보를 조회할 수 있도록 구현
- API 를 통해 받아온 수질 정보에 대응하여 직관적으로 사용자가 적합/부적합 여부를
- 확인할 수 있는 이미지 연동

팀원 김동현

- 각 제보게시글의 추천수를 바탕으로 랭킹을 산정하는 기능 구현
- 제보게시글이 작성된 지역정보를 바탕으로 제보가 많은 지역의 랭킹을 산정하는 기능을 구현

03

프로젝트상세

3-1. ERD

3-2. UML

3-3. 주요 기능 상세 설명

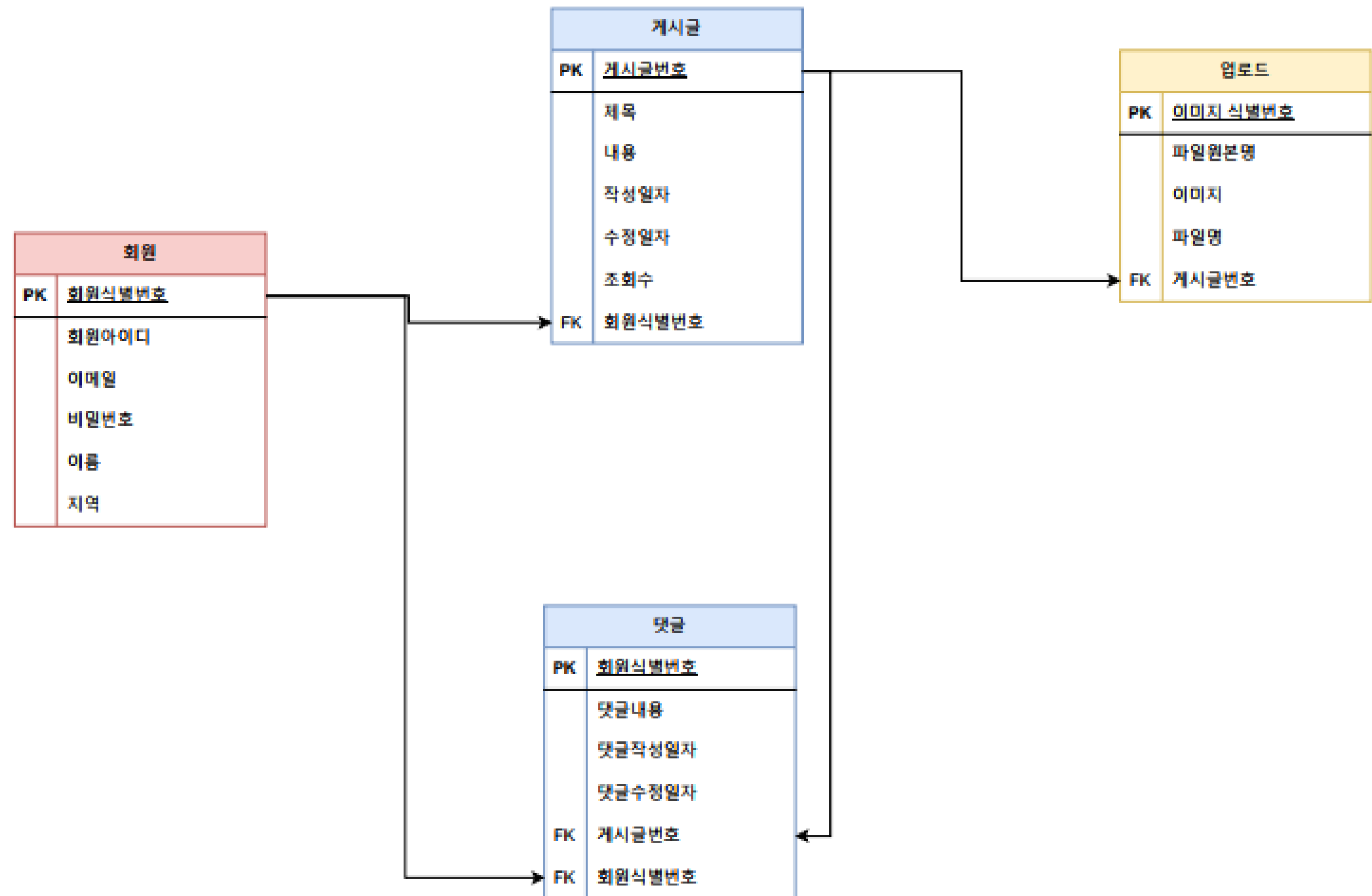
03 프로젝트상세

3-1. ERD

3-2. UML

3-3. 주요기능 상세설명

ERD



03 프로젝트상세

3-1. ERD

3-2. UML

3-3. 주요기능 상세설명

UML



기능 상세설명 (메인 페이지 Api 구현)

메인 페이지 API 화면 / select.html

날짜 선택

2024-07-02

지역 선택

거제시

검색

pH

적합 기준 : 5.8~8.5

탁도 (NTU)

적합 기준 : 0.5이하

잔류염소 (ml/L)

적합 기준 : 0.1~4.0

메인 페이지 화면에서 날짜, 지역 선택 후 검색 클릭 시
ApiController로 이동하여 선택에 맞는 API 정보를 불러온다

날짜 선택

2024-07-02

지역 선택

거제시

검색

pH

6.86

적합 기준 : 5.8~8.5

탁도 (NTU)

0.004

적합 기준 : 0.5이하

잔류염소 (ml/L)

0.70

적합 기준 : 0.1~4.0

선택한 지역, 날짜에 대한 API값이 화면에 출력, 값에 따라 이모티콘 변경

[ApiController.java]

```
@PostMapping("/main")
public String submitSelectForm(@ModelAttribute ApiRequestParams requestParams, Model model) {
    System.out.println("post");
    requestParams.setEddt(requestParams.getStdtd());

    String apiUrl = "http://opendata.kwater.or.kr/openapi-data/service/pubd/waterinfos/waterquality/daywater/list?" +
        "servicekey=0yknVZHJ5V4d1L6T09qIKVxU5jf3hqmPaEt%2FfiGXoddQEfQwRC56sRqh0dmcPJ6U8Jphw4kG2EE%2F1HisQc8tA%3D%3D" +
        "&_type=json" +
        "&numOfRows=1" +
        "&pageNo=1" +
        "&sgcCd=" + requestParams.getSgcCd() +
        "&stdt=" + requestParams.getStdtd() +
        "&eddt=" + requestParams.getEddt();

    RestTemplate restTemplate = new RestTemplate();
    String jsonResponse = restTemplate.getForObject(apiUrl, String.class);

    ObjectMapper objectMapper = new ObjectMapper();
    try {
        JsonNode rootNode = objectMapper.readTree(jsonResponse);
        JsonNode bodyNode = rootNode.path("response").path("body");
        JsonNode itemNode = bodyNode.path("items").path("item");

        String pH = itemNode.path("data4").asText();
        String turbidity = itemNode.path("data5").asText();
        String residualChlorine = itemNode.path("data6").asText();

        model.addAttribute("pH", pH);
        model.addAttribute("turbidity", turbidity);
        model.addAttribute("residualChlorine", residualChlorine);
        model.addAttribute("requestParams", requestParams);
    } catch (JsonProcessingException e) {
        e.printStackTrace();
    }

    model.addAttribute("regions", regions);
    return "select";
}
```

외부 API를 받아오기 위한 URL 생성,
받아온 API중에 입력 받은 데이터를 추출, 모델에 추가
select.html 파일에 전달

기능 상세설명 (메인 페이지 추천 게시물/최다 제보지역 구현화면)

메인 페이지 화면 / select.html

추천 게시물

순위	제목	작성자	추천수
1	우리동네 정수장 수질 현황 - 39	user9	5
2	우리동네 정수장 수질 현황 - 40	user10	5
3	우리동네 정수장 수질 현황 - 25	user5	5

최다 제보지역

순위	지역	제보 수
1	통화군	10
2	사천시	10
3	동두천시	10

```
<table class="table table-hover">
  <h2>추천 게시물</h2>
  <br>
  <thead class="table-primary">
    <tr>
      <th>순위</th>
      <th>제목</th>
      <th>작성자</th>
      <th>추천수</th>
    </tr>
  </thead>
  <tbody>
    <tr th:each="board, iterStat : ${top3RecommendedBoards}">
      <td th:text="${iterStat.index + 1}"></td>
      <td>
        <a th:href="@{'/board/read/' + ${board.boardNum}}" class="no-underline">
          <h5 th:text="${board.title}"></h5>
        </a>
      </td>
      <td>
        <p th:text="${board.user.name}"></p>
      </td>
      <td>
        <p th:text="${#lists.size(board.recommender)}"></p>
      </td>
    </tr>
  </tbody>
</table>
```

추천 게시물에서 제목을 클릭 시 a태그에 작성된
http://localhost:8081/board/read /\${board.boardNum} 주소로 이동
선택한 게시글의 화면을 보여준다

ApiController.java

```
@Autowired
private BoardService boardService;

@GetMapping("/main")
public String showSelectForm(Model model) {
    List<Board> top3RecommendedBoards = boardService.findTop3RecommendedBoards();
    model.addAttribute("top3RecommendedBoards", top3RecommendedBoards);

    List<Object[]> top3ActiveRegions = boardService.findTop3ActiveRegions();
    List<String> activeRegions = new ArrayList<>();
    List<Long> numPosts = new ArrayList<>();

    for (Object[] result : top3ActiveRegions) {
        String regiona = (String) result[0];
        Long counta = (Long) result[1];
        activeRegions.add(regiona);
        numPosts.add(counta);
    }

    model.addAttribute("activeRegions", activeRegions);
    model.addAttribute("numPosts", numPosts);

    return "select";
}
```

BoardService의 메소드를 호출해서
상위 3개의 게시글을 가져오는 쿼리를 실행, 메인페이지에 보여준다

BoardRepository.java

```
@Query(value = "SELECT b.* " +
  "FROM Board b " +
  "JOIN (SELECT BOARD_BOARD_NUM, COUNT(*) AS RECOMMEND_COUNT " +
  "      FROM BOARD_RECOMMENDER " +
  "      GROUP BY BOARD_BOARD_NUM " +
  "      ORDER BY RECOMMEND_COUNT DESC " +
  "      LIMIT 3) br ON b.board_num = br.BOARD_BOARD_NUM",
  nativeQuery = true)
List<Board> findTop3Recommender();

@Query(value = "SELECT u.region, COUNT(b.board_num) AS num_posts " +
  "FROM Board b " +
  "JOIN Site_User u ON b.user_id = u.id " +
  "GROUP BY u.region " +
  "ORDER BY num_posts DESC " +
  "LIMIT 3",
  nativeQuery = true)
List<Object[]> findTop3Regions();
```

Native SQL 쿼리를 사용하여 메인 게시물 중
추천을 가장 많이 받은 게시물 3개와
게시글이 가장 많은 지역 3개를 조회한다

BoardService.java


```
public List<Board> findTop3RecommendedBoards() {
    return boardRepository.findTop3Recommender();
}

public List<Object[]> findTop3ActiveRegions() {
    return boardRepository.findTop3Regions();
}
```

BoardRepository의 쿼리 메소드를 호출하여
추천 게시물과 지역 정보를 가져온다

기능 상세설명 (게시글 추천 기능)

게시글 조회 화면 / boardRead.html



추천인 ID :
user7 user4 user5

```
<div class = "m-2">
  <a href="javascript:void(0);" class="recommend" th:data-uri="@{/board/recommend/${board.boardNum}}">
    <span class="badge rounded-pill bg-danger"
      th:text="${#lists.size(board.recommender)}"></span>
    </a>

  <ul id="recommender-list" >
    <br> 추천인 ID :
    <br>
    <li th:each="user : ${board.recommender}" th:text="${user.loginId}" class="recommender-list"></li>
  </ul>
</div>
```

‘게시글 조회’ 화면에서 추천 이미지를 클릭 아래 스트립트에 작성된 함수가 동작

localhost:8081 내용:

정말로 추천하시겠습니까? (이미 추천을 했다면, 추천이 취소됩니다.)

확인

취소

```
<script>
  document.addEventListener("DOMContentLoaded", function() {
    const recommend_elements = document.getElementsByClassName("recommend");
    Array.from(recommend_elements).forEach(function(element) {
      element.addEventListener('click', function() {
        if (confirm("정말로 추천하시겠습니까? (이미 추천을 했다면, 추천이 취소됩니다.)")) {
          location.href = this.dataset.uri;
        }
      });
    });
  });
</script>
```

추천을 확인하는 메시지 출력 후 확인을 누르면,
a태그에 작성된 localhost:8080/board/recommend/\${board.boardNum} 주소로 이동

BoardController.java

```
@PreAuthorize("isAuthenticated()")
@GetMapping("/recommend/{id}")
public String boardRecommend(Principal principal, @PathVariable("id") Integer id) {
    Board board = this.boardService.getBoardById(id);
    SiteUser siteUser = this.userService.getUser(principal.getName());
    this.boardService.recommend(board, siteUser);
    return String.format("redirect:/board/read/%s", id);
}
```

BoardService에 작성된 recommend 함수를 작동하여 추천/ 추천 취소 동작을 가능하게 함.
작업이 완료되면 해당 게시글의 주소로 이동

BoardService.java


```
public void recommend(Board board, SiteUser siteUser) {
    Set<SiteUser> recommenders = board.getRecommender();
    if (recommenders.contains(siteUser)) {
        recommenders.remove(siteUser);
    } else {
        recommenders.add(siteUser);
    }
    this.boardRepository.save(board);
}
```

해당 게시글의 추천 목록(recommender)에 추천을 누른 SiteUser객체가 이미 존재한다면 목록에서 제거(추천 취소), 존재하지 않는다면 목록에 추가(추천)

게시글 조회 화면

조회수: 6

추천수: 4



추천인 ID :
user5 user4 user1 user7

기능 상세설명 (회원가입 시 유효성 검사)

회원가입

ID

비밀번호

비밀번호 확인

이름

이메일

이메일 인증

인증 코드

인증 코드 확인

지역

거제시

회원가입

UserCreateForm.java

```
@Getter
@Setter
public class UserCreateForm {

    @NotEmpty(message = "사용자 ID는 필수 입력 항목입니다.")
    private String loginId;

    @NotEmpty(message = "비밀번호는 필수 입력 항목입니다.")
    private String password1;

    @NotEmpty(message = "비밀번호 확인은 필수 입력 항목입니다.")
    private String password2;

    @NotEmpty(message = "이름은 필수 입력 항목입니다.")
    private String name;

    @NotEmpty(message = "이메일은 필수 입력 항목입니다.")
    @Email
    private String email;
```

UserCreateForm을 통해 NotEmpty annotation을 이용하여 필수 입력 항목 값들에 대한 경고문구 처리

이메일

지역

이메일

지역

이메일

지역

회원가입

비밀번호는 필수 입력 항목입니다.
비밀번호 확인은 필수 입력 항목입니다.
사용자 ID는 필수 입력 항목입니다.
이메일은 필수 입력 항목입니다.
이름은 필수 입력 항목입니다.

ID

비밀번호

비밀번호 확인

이름

이메일

localhost:8081 내용:

비밀번호를 다시 한번 확인해주세요.

확인

비밀번호 확인이 제대로 처리되지 않은 상태에서 회원가입을 하려고 할 때의 알림창 출력 화면

signup.html

```
function pwCheck() {
    var pw1 = $('#pw1').val();
    var pw2 = $('#pw2').val();

    if (pw1 != '' && pw2 != '' && pw1 != pw2) {
        $('#pwConfirm').text('비밀번호가 일치하지 않습니다.').css('color', 'red');
        $('#pw2').css('border-color', 'red');
        return false;
    } else {
        $('#pwConfirm').text('비밀번호가 일치합니다.').css('color', 'green');
        $('#pw2').css('border-color', 'green');
        return true;
    }
}
```

비밀번호

비밀번호 확인

비밀번호가 일치하지 않습니다.

비밀번호란과 비밀번호 확인란에 입력한 비밀번호가 일치하지 않을 시 알림 메시지 출력

```
$('#submitBtn').click(function() {
    if (!emailVerified) {
        alert('회원가입을 위해 이메일 인증이 필요합니다.');
```

localhost:8081 내용:

회원가입을 위해 이메일 인증이 필요합니다.

확인

이메일 인증 절차를 거치지 않은 상태에서 회원가입 버튼을 누를 시 알림창

기능 상세설명 (회원가입 시 이메일 인증 - 인증번호 발송)

[회원가입 화면 / signup.html]

이메일

qlqj47@naver.com

이메일 인증

```
<input type="email" id="email" th:field="*{email}" class="form-control">
<button type="button" class="btn" id="sendVerificationEmail">이메일 인증</button>
```

```
$('#sendVerificationEmail').click(function() {
    var email = $('#email').val();
    if (email) {
        $.ajax({
            type: 'POST',
            url: '/user/send-email',
            contentType: 'application/json',
            data: JSON.stringify({ 'email': email }),
            success: function(response) {
                alert('이메일이 발송되었습니다.');
```

회원가입 화면의 이메일 인증 버튼을 클릭하면 같은 파일 내 스크립트 코드의 함수가 동작

[EmailController]

```
@PostMapping("/send-email")
@ResponseBody
public String sendEmail(@RequestBody EmailRequest emailRequest) throws MessagingException {
    String email = emailRequest.getEmail();
    String verificationCode = generateVerificationCode();
    verificationCodes.put(email, verificationCode);
    emailService.sendAuthenticationEmail(email, verificationCode);
    return "이메일이 발송되었습니다.";
}
```

임의로 생성한 8자리 랜덤코드와 이메일 주소를 추후 인증과정에 사용하기 위해 verificationCodes에 저장해주고 EmailService 내의 sendAuthenticationEmail 함수를 호출하여 인증코드를 입력 받은 이메일 주소로 발송

```
// 8자리 랜덤 코드
private String generateVerificationCode() {
    final String CHARACTERS = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789";
    final int LENGTH = 8;
    final Random random = new Random();

    StringBuilder sb = new StringBuilder();

    for (int i = 0; i < LENGTH; i++) {
        int index = random.nextInt(CHARACTERS.length());
        sb.append(CHARACTERS.charAt(index));
    }

    return sb.toString();
}
```

8자리 랜덤코드를 생성하여 인증번호로 사용

기능 상세설명 (회원가입 시 이메일 인증 - 인증번호 발송)

[EmailService]

```
@Service
public class EmailSendingService {

    @Autowired
    private JavaMailSender javaMailSender;

    public void sendAuthenticationEmail(String recipientEmail, String authenticationCode) throws MessagingException {
        MimeMessage message = javaMailSender.createMimeMessage();
        MimeMessageHelper helper = new MimeMessageHelper(message, true, "UTF-8");

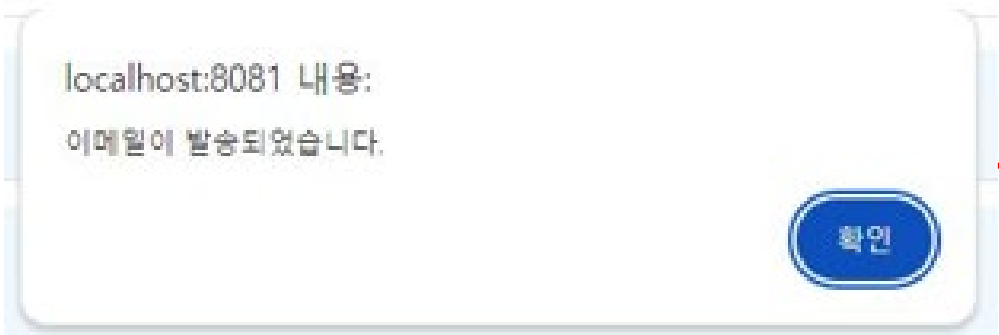
        String subject = "지방상수도 수질 정보 제공 페이지 이메일 인증";

        String htmlContent = "<html><body>" +
            "<div style='background-color: #87CEEB; padding: 20px;'>" +
            "<div style='background-color: white; padding: 20px; border-radius: 10px;'>" +
            "<h2 style='color: #32CD32; text-align: center;'>지방상수도 수질 정보 제공 페이지 코드</h2>" +
            "<p style='text-align: center;'>아래 코드를 입력하여 이메일 인증을 완료해주세요</p><br><br>" +
            "<div style='text-align: center; border: 2px solid #32CD32; padding: 10px; font-size: 18px; font-weight: bold;'>" +
            authenticationCode +
            "</div>" +
            "<br><br><p style='text-align: center;'>이메일 인증을 위해 위 코드를 입력하면 회원가입이 정상 진행됩니다.</p>" +
            "</div></div></body></html>";

        helper.setTo(recipientEmail);
        helper.setSubject(subject);
        helper.setText(htmlContent, true);

        javaMailSender.send(message);
    }
}
```

JavamailSender, MimeMessage, MimeMessageHelper를 사용하여 인증코드를 보낼 이메일의 양식을 입력하여 메시지를 전송

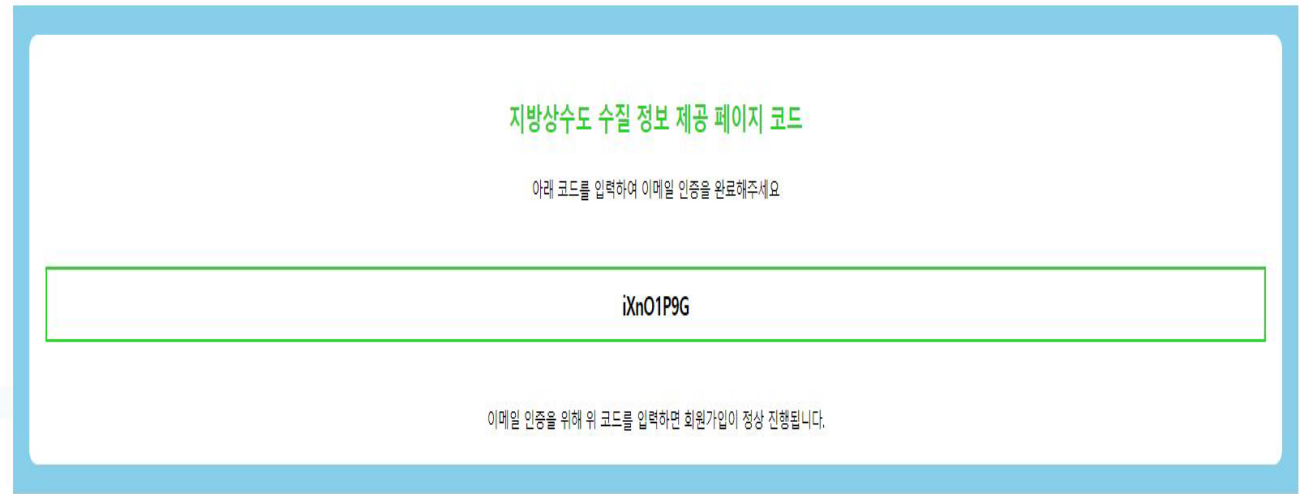


[build.gradle]

```
// 스프링 부트에서 이메일 기능을 쉽게 통합하고 사용할 수 있도록 해주는 스타터,  
// 이를 통해 JavaMailSender를 사용하여 이메일을 보내는 작업을 단순화  
implementation 'org.springframework.boot:spring-boot-starter-mail'
```

[Application Properties 설정]

```
# Email 인증 관련 설정  
spring.mail.host=smtp.gmail.com  
spring.mail.port=587  
spring.mail.username=본인 이메일 주소 입력  
spring.mail.password=본인 gmail 앱 비밀번호 입력  
spring.mail.properties.mail.smtp.auth=true  
spring.mail.properties.mail.smtp.starttls.enable=true
```



기능 상세설명 (회원가입 시 이메일 인증 - 인증번호 검증)

[회원가입 화면 / signup.html]

인증 코드

인증 코드 확인

```
<input type="text" id="emailCode" class="form-control">
<button type="button" class="btn" id="verifyEmailCode">인증 코드 확인</button>
<div id="codeVerificationResult"></div>
```

```
var emailVerified = false;

$('#verifyEmailCode').click(function() {
    var email = $('#email').val();
    var code = $('#emailCode').val();

    $.ajax({
        type: 'POST',
        url: '/user/verify-code',
        contentType: 'application/json',
        data: JSON.stringify({ email: email, code: code }),
        success: function(response) {
            if (response.success) {
                $('#codeVerificationResult').html('<div class="alert alert-success">' + response.message + '</div>');
                emailVerified = true;
            } else {
                $('#codeVerificationResult').html('<div class="alert alert-danger">' + response.message + '</div>');
                emailVerified = false;
            }
        },
        error: function() {
            $('#codeVerificationResult').html('<div class="alert alert-danger">서버 오류가 발생했습니다. 잠시 후 다시 시도해주세요.</div>');
            emailVerified = false;
        }
    });
});
```

회원가입 화면의 인증 코드 확인 버튼을 클릭하면 같은 파일 내 스크립트 코드의 함수가 동작

emailVerified 변수는 인증 코드 확인 절차를 거쳤는 지 판단하여 최종 회원 가입 시 인증 절차를 통과해야만 회원가입이 가능 하도록 유효성 검사를 실시하는 근거로 사용

[EmailController]

```
@PostMapping("/verify-code")
@ResponseBody
public Map<String, Object> verifyCode(@RequestBody VerificationRequest verificationRequest) {
    String email = verificationRequest.getEmail();
    String code = verificationRequest.getCode();
    Map<String, Object> response = new HashMap<>();

    String savedCode = verificationCodes.get(email);
    if (savedCode != null && savedCode.equals(code)) {
        response.put("success", true);
        response.put("message", "인증 코드가 확인되었습니다.");
        verificationCodes.remove(email);
    } else {
        response.put("success", false);
        response.put("message", "잘못된 인증 코드입니다.");
    }
    return response;
}
```

인증번호 발송 단계에서 저장해주었던 이메일과 인증코드를 바탕으로 현재 검증을 실시하려는 이메일과 인증코드를 비교하여 인증 여부(success)와 인증 결과에 따른 출력 메시지(message)를 Map 형태로 반환

[회원가입 화면 / signup.html]

이메일

qlqj47@naver.com

이메일 인증

인증 코드

IXnO1P9G

인증 코드가 확인되었습니다.

이메일

qlqj47@naver.com

이메일 인증

인증 코드

ixn01p9g

잘못된 인증 코드입니다.

인증 성공

인증 실패

기능 상세설명 (게시글 업로드 기능 - 1)

이미지:



```
try {
    List<UploadDTO> uploadDTOs = new ArrayList<>();
    for (MultipartFile file : files) {
        try {
            UploadDTO uploadDTO = this.uploadService.uploads(file);
```

[boardController.java]

MultipartFile 인터페이스 방식을 통해 file 변수 선언 후 uploadDTO 객체를 ArrayList 배열로 uploadDTOs에 담은 후 uploadService에서 uploads 메소드 실행

```
        uploadDTOs.add(uploadDTO);
    } catch (IOException e) {
        e.printStackTrace();
        redirectAttributes.addFlashAttribute("errorMessage", "파일 업로드 중 오류가 발생하였습니다.");
        return "redirect:/board/list";
    }

    boardService.create(boardForm.getTitle(), boardForm.getBoardContent(), siteUser, uploadDTOs);

} catch (Exception e) {
    e.printStackTrace();
    redirectAttributes.addFlashAttribute("errorMessage", "게시글 작성 중 오류가 발생하였습니다.");
    return "redirect:/board/list";
}

return "redirect:/board/list";
}
```

[boardController.java]

반환된 UUID, 파일명, 이미지로 uploadDTO 객체에 추가하여 boardService의 create 메소드로 전달

```
project.environment.upload.path=./src/main/resources/static/images
```

[application.properties]

```
package project.environment.service;

import net.coobird.thumbnailator.Thumbnailator;

@Service
public class UploadService {

    @Value("${project.environment.upload.path}")
    private String uploadPath;

    public UploadDTO uploads(MultipartFile file) throws IOException {
        String originalName = file.getOriginalFilename();
        String uuid = UUID.randomUUID().toString();
        String saveName = uuid + "_" + originalName;
        Path savePath = Paths.get(uploadPath, saveName);

        file.transferTo(savePath);

        String contentType = Files.probeContentType(savePath);
        boolean isImage = contentType != null && contentType.startsWith("image");

        if (isImage) {
            File thumbnailFile = new File(uploadPath, "s_" + saveName);
            Thumbnailator.createThumbnail(savePath.toFile(), thumbnailFile, 200, 200);
        }

        UploadDTO uploadDTO = new UploadDTO();
        uploadDTO.setUuid(uuid);
        uploadDTO.setFileName(saveName);
        uploadDTO.setImg(isImage);

        return uploadDTO;
    }
}
```

[UploadService.java]

@Value로 application.properties 파일 경로 에너테이션 설정

MultipartFile을 통한 file을 전달 받아 파일 기존 이름을 얻은 후 UUID 고유번호 생성 후 파일명을 uuid_업로드파일명으로 호출하여 실제 파일을 저장

uploadDTO 객체에 UUID, 저장된 파일명, 이미지 여부를 설정 후 반환

기능 상세설명 (게시글 업로드 기능 - 2)

```
public void create(String title, String boardContent, SiteUser user, List<UploadDTO> uploadDTOs) {
    Board board = new Board();
    board.setTitle(title);
    board.setBoardContent(boardContent);
    board.setUser(user);
    board.setHitCount(0L);
    board.setCreateDate(LocalDate.now());

    List<Upload> uploads = uploadDTOs.stream()
        .map(uploadDTO -> {
            Upload upload = Upload.fromDTO(uploadDTO);
            upload.setBoard(board);
            return upload;
        })
        .collect(Collectors.toList());

    board.setUploads(uploads);

    boardRepository.save(board);
}
```

[boardService.java]

게시글 생성에 필요한 제목, 내용, 작성자를 받아 board 객체에 set으로 입력
이전 전달받은 uploadDTOs 리스트를 스트림으로 변환

uploadDTO 객체를 기반으로 upload 엔티티를 생성한 후에 게시글에 속하는 번호 설정

변환된 upload 엔티티들을 리스트로 수집 후 게시글 객체에 변환된 리스트 저장

[Upload.java]

```
@Entity
@Getter
@Setter
@ToString
public class Upload {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column
    private String uuid;

    @Column
    private String fileName;

    @Column
    private boolean img;

    @ManyToOne
    @JoinColumn(name = "board_id")
    private Board board;

    public static Upload fromDTO(UploadDTO uploadDTO) {
        Upload upload = new Upload();
        upload.setFileName(uploadDTO.getFileName());
        upload.setUuid(uploadDTO.getUuid());
        upload.setImg(uploadDTO.isImg());
        return upload;
    }

    public void setBoard(Board board) {
        this.board = board;
    }
}
```

고유의 값(UUID), 파일명, img(업로드 유무),
매핑된 board로 구성

기능 상세설명 (마이페이지)

UserRepository.java

```
public interface UserRepository extends JpaRepository<SiteUser, Long>{
    Optional<SiteUser> findByLoginId(String loginId);
}
```

BoardRepository.java

```
@Repository
public interface BoardRepository extends JpaRepository<Board, Integer> {
    Page<Board> findAll(Pageable pageable);
    List<Board> findByUser(SiteUser user);
}
```

CommentsRepository.java

```
@Repository
public interface CommentsRepository extends JpaRepository<Comments, Long> {
    List<Comments> findByUser(SiteUser user);
}
```

각각의 회원 별 작성한 게시글과,댓글을
데이터베이스에서 가져옴

UserService.java

```
public SiteUser getUser(String loginId) {
    Optional<SiteUser> siteUser = this.userRepository.findByLoginId(loginId);

    if(siteUser.isPresent()) {
        return siteUser.get();
    }else{
        throw new DataNotFoundException("siteuser not found");
    }
}
```

BoardService.java

```
public List<Board> getBoardsByUser(SiteUser user) {
    return boardRepository.findByUser(user);
}
```

CommentsService.java

```
public List<Comments> getBoardsByUser(SiteUser user) {
    return commentsRepository.findByUser(user);
}
```

MyPageController.java

```
@RequiredArgsConstructor
@Controller
public class MyPageController {

    private final UserService userService;
    private final BoardService boardService;
    private final CommentsService commentsService;

    @GetMapping("/mypage")
    public String myPage(Model model, Principal principal) {
        if (principal == null) {
            return "redirect:/user/login";
        }

        String loginId = principal.getName();
        SiteUser user = userService.getUser(loginId);

        List<Board> boards = boardService.getBoardsByUser(user);

        List<Comments> comments = commentsService.getBoardsByUser(user);

        model.addAttribute("user", user);
        model.addAttribute("boards", boards);
        model.addAttribute("comments", comments);

        return "mypage";
    }
}
```

마이페이지에 자신이 작성한
게시글과,댓글을
목록을 볼 수 있게 구현

내 정보

아이디: user1
이메일: user1@gmail.com
[정보수정](#)

[작성한 게시글](#) [작성한 댓글](#)

내가 작성한 게시글

게시글 번호	제목	작성일자
1	우리동네 정수장 수질 현황 - 1	2024-07-02
11	우리동네 정수장 수질 현황 - 11	2024-07-02
21	우리동네 정수장 수질 현황 - 21	2024-07-02
31	우리동네 정수장 수질 현황 - 31	2024-07-02
41	우리동네 정수장 수질 현황 - 41	2024-07-02
51	우리동네 정수장 수질 현황 - 51	2024-07-02
61	우리동네 정수장 수질 현황 - 61	2024-07-02
71	우리동네 정수장 수질 현황 - 71	2024-07-02
81	우리동네 정수장 수질 현황 - 81	2024-07-02
91	우리동네 정수장 수질 현황 - 91	2024-07-02

04

Trouble Shooting 및 개발리뷰

4-1. Trouble Shooting

4-2. 개발리뷰

Trouble Shooting (1. 추천 기능 작업 시 Javascript alert 동작 오류 해결)

문제 : 기능 구현을 위해 추천 이미지 버튼에 recommend라는 클래스 name을 붙이고 recommend의 클래스 name을 가진 요소에 대해서 클릭했을 때, "정말로 추천하시겠습니까?"라는 확인 알림 창이 뜨고 확인 버튼을 눌렀을 때 이와 관련된 Controller로 이동하는 로직을 구성하였으나 화면에서 버튼을 클릭했을 때 알림창이 아예 나타나지 않는 문제가 발생

해결 : 이 문제는 JavaScript코드가 DOM이 완전히 로드되기 전에 실행되기 때문일 가능성이 높다고 판단해, JavaScript 코드를 DOMContentLoaded 이벤트 안에 넣어 주어서 DOM이 완전히 로드된 후에 이벤트 리스너가 설정되도록 코드를 수정함. 이후 알림창(추천을 확인하는 대화 상자)가 정상적으로 출력됨.

BoardRead.html의 추천 버튼을 눌렀을 때의 동작과 관련한 Script 코드

```
<script>
  const recommend_elements =
  document.getElementsByClassName("recommend");
  Array.from(recommend_elements).forEach(function(element) {
    element.addEventListener('click', function() {
      if(confirm("정말로 추천하시겠습니까?")) {
        location.href = this.dataset.uri;
      };
    });
  });
</script>
```

```
<script>
  Document.addEventListener("DOMContentLoaded", function() {
    const recommend_elements =
    document.getElementsByClassName("recommend");
    Array.from(recommend_elements).forEach(function(element) {
      element.addEventListener('click', function() {
        if(confirm("정말로 추천하시겠습니까?")) {
          location.href = this.dataset.uri;
        };
      });
    });
  });
</script>
```

Trouble Shooting (2. 게시물에 이미지 업로드 시 조회 화면에 바로 반영되지 않던 문제)

문제 : 게시물 작성시에 이미지를 업로드하는 기능에 대해서 BoardService의 create 함수에서 업로드에 사용한 이미지 파일의 파일명과 이 파일명을 바탕으로 파일의 경로를 저장하는 변수를 만들어 저장하고 조회 시 이 경로를 바탕으로 표현하는 방식으로 단순히 로직을 작성하였다. 이 때, 디렉토리 리소스를 웹 리소스로 핸들링 해주는 config를 작성했음에도 불구하고 업로드한 이미지가 조회화면에서 서버를 재시작 하지 않으면 조회가 불가능한 상태였다.

```
public Board create(Board board, @RequestParam(name="file", required=false) MultipartFile file) throws Exception {
```

```
    String projectPath = System.getProperty("user.dir")
        + "\\src\\main\\resources\\static\\files";
```

```
    UUID uuid = UUID.randomUUID();
    String fileName = uuid + "_" + file.getOriginalFilename();
    File saveFile = new File(projectPath, fileName);
    file.transferTo(saveFile);
```

```
    board.setFilename(fileName);
    board.setFilepath("/files/" + fileName);
    board.setCreate_Date(LocalDate.now());
    board.setModify_Date(null);
    board.setHit_Count(0L);
```

```
    return boardRepository.save(board);
```

```
}
```

해결 : 본래 시도했던 방식 자체의 한계가 있음을 알고, 이미지를 업로드 하는 방식 자체를 전체적으로 달리하여 진행하였다.

이미지를 처리하는 데 필요한 정보를 DTO를 통해 별도로 데이터를 묶어서 처리하는 방식으로 구현했으며, 기존 파일명과 이미지파일명에 대해서도 board 객체가 아닌 별도 Upload 객체를 생성하여 board에서의 이미지 업로드 정보에 대해 저장하는 방식으로 변경하였다.

```
@Value("${project.environment.upload.path}")
```

```
private String uploadPath;
```

```
public UploadDTO uploads(MultipartFile file) throws IOException {
```

```
    String originalName = file.getOriginalFilename();
```

```
    String uuid = UUID.randomUUID().toString();
```

```
    String saveName = uuid + "_" + originalName;
```

```
    Path savePath = Paths.get(uploadPath, saveName);
```

```
    file.transferTo(savePath);
```

```
    String contentType = Files.probeContentType(savePath);
```

```
    boolean isImage = contentType != null && contentType.startsWith("image");
```

```
    if (isImage) {
```

```
        File thumbnailFile = new File(uploadPath, "s_" + saveName);
```

```
        Thumbnailator.createThumbnail(savePath.toFile(), thumbnailFile, 200, 200);
```

```
    }
```

```
    UploadDTO uploadDTO = new UploadDTO();
```

```
    uploadDTO.setUuid(uuid);
```

```
    uploadDTO.setFileName(saveName);
```

```
    uploadDTO.setImg(isImage);
```

```
    return uploadDTO;
```

```
}
```

Trouble Shooting (3. Board/SiteUser Entity를 바탕으로 랭킹을 산정하는 방식과 관련한 문제)

문제 : 게시글(Board)의 클래스에는 게시글을 작성한 사용자(SiteUser) 클래스를 변수로 가지고 있고 사용자의 지역정보를 바탕으로 합계 또는 순위 비교를 하는 과정에서 DB상 한 테이블에 있는 정보만을 사용하는 것이 아니어서 우리가 가진 정보를 가공하여 원하는 결과물을 출력하는 방식을 기획하는 단계에서 고민이 있었다.

추천 게시글

순위	제목	작성자	추천수
1	우리동네 정수장 수질 현황 - 13	user3	7
2	우리동네 정수장 수질 현황 - 53	user3	6
3	우리동네 정수장 수질 현황 - 25	user5	5

최다 제보지역

순위	지역	제보 수
1	봉화군	20
2	사천시	10
3	예천군	10

해결 : 이 문제는 다른 방식으로도 여러 해결이 가능하지만, 우리는 이미 정보를 가지고 있는 Board 클래스(엔티티)를 이용해서 JPA에 직접 SQL문을 작성하여 원하는 방식으로 레퍼지토리 내에서 처리가능하도록 작업

```
@Repository
public interface BoardRepository extends JpaRepository<Board, Integer> {
    Page<Board> findAll(Pageable pageable);
    List<Board> findByUser(SiteUser user);

    @Query(value = "SELECT b.* " +
        "FROM Board b " +
        "JOIN (SELECT BOARD_BOARD_NUM, COUNT(*) AS RECOMMEND_COUNT " +
        "      FROM BOARD_RECOMMENDER " +
        "      GROUP BY BOARD_BOARD_NUM " +
        "      ORDER BY RECOMMEND_COUNT DESC " +
        "      LIMIT 3) br ON b.board_num = br.BOARD_BOARD_NUM",
        nativeQuery = true)
    List<Board> findTop3Recommender();

    @Query(value = "SELECT u.region, COUNT(b.board_num) AS num_posts " +
        "FROM Board b " +
        "JOIN Site_User u ON b.user_id = u.id " +
        "GROUP BY u.region " +
        "ORDER BY num_posts DESC " +
        "LIMIT 3",
        nativeQuery = true)
    List<Object[]> findTop3Regions();
}
```

04 프로젝트시연

4-1. Trouble Shooting

4-2. 개발리뷰

개발리뷰

아쉬웠던 점	잘했다고 생각하는 점
<p>초기 프로젝트의 기능 개발 단계에서 빈번하게 에러가 나타나고 이를 수정하는 과정이 많았다. 에러들을 처리하는 과정에서 <u>thymeleaf 문법을 제대로 적용하지 못해서 발생하는 문제</u>가 다수 있었다는 것을 알게 되었고, 학원 교육 과정에서는 자세히 다루지는 않았지만 <u>주로 사용되는 thymeleaf 기본 문법을 숙지</u>하게 됨에 따라 관련 error들이 발생하는 일이 줄어들었다.</p> <p>Entity들을 정의하는 단계에서 JPA를 통해 데이터베이스에서 관리될 컬럼명을 입력하는 과정 중 DB에서 컬럼명이 표현되는 방식과 혼동하여 Entity 중 컬럼명에 해당할 <u>변수명의 첫글자를 대문자로 입력하는 실수</u>가 있었다. 프로젝트 개발 진행 과정 중 이를 인지하고, 변수명의 첫 글자를 소문자로 바꿔주고 이에 대응하는 변수명을 수정하는 다소 소모적인 작업이 있었다. 사소하지만 기본적인 규칙을 잘 지켜야 함을 몸소 깨닫고 후에 기능 개선 과정 중 추가적으로 다루지는 변수명들을 작업하는 과정에서는 이를 특히 신경 써서 작업하였다.</p>	<p>실제 운영중인 커뮤니티를 보면 게시글 목록 중 작성자의 이름을 클릭하면 작성자의 게시글이나 댓글목록을 확인할 수 있는 창으로 이동 가능하게 설정한 점, 비로그인 상태에서는 댓글 작성창이 보이지만 비활성화해두되 로그인 시 작성 가능함을 알려주는 안내를 작성한 점 등과 같은 부분에서 이번 프로젝트를 진행하는 과정에서는 좀 더 <u>사용자의 접근성 관점에서 접근해보려고 노력</u>했다.</p> <p>프로젝트 진행 과정 중 상당히 중요하다고 볼 수 있는 개발진행 단계에서 조원 중 한 명이 예비군 훈련 참석으로 부득이하게 4일을 결석한 일이 있었다. 이 과정에서 매일 현재 프로젝트의 진행 상황과 명일 작업 예정 내용에 대한 일지를 작성하고 노션 및 카카오톡으로 공유를 함으로써 조원이 복귀했을 때 프로젝트의 진행상황에 대해 빠르게 이해하고 상황에 맞는 기능 구현 역할을 부여해서 같이 프로젝트를 잘 마무리 할 수 있었다. 이런 과정에서 <u>커뮤니케이션 능력에 강점이 있다</u>고 생각한다.</p> <p>프로젝트 중간에 <u>예상치 못한 기술적 문제가 발생했을 때, 해결방안에 여러 접근법들을 주저함이 없이 사용</u>했던 점이 좋았다. 비슷한 문제에 직면했던 다른 조 조원들에게 도움을 요청하기도, 강사님/조교님과의 피드백을 적극적으로 하고, 온라인 상에 제공되는 블로그나 문서 정보들을 활용하기도 하고, Open AI를 활용하기도 하면서 문제상황을 적극적으로 해결하였다.</p>

Thank you
