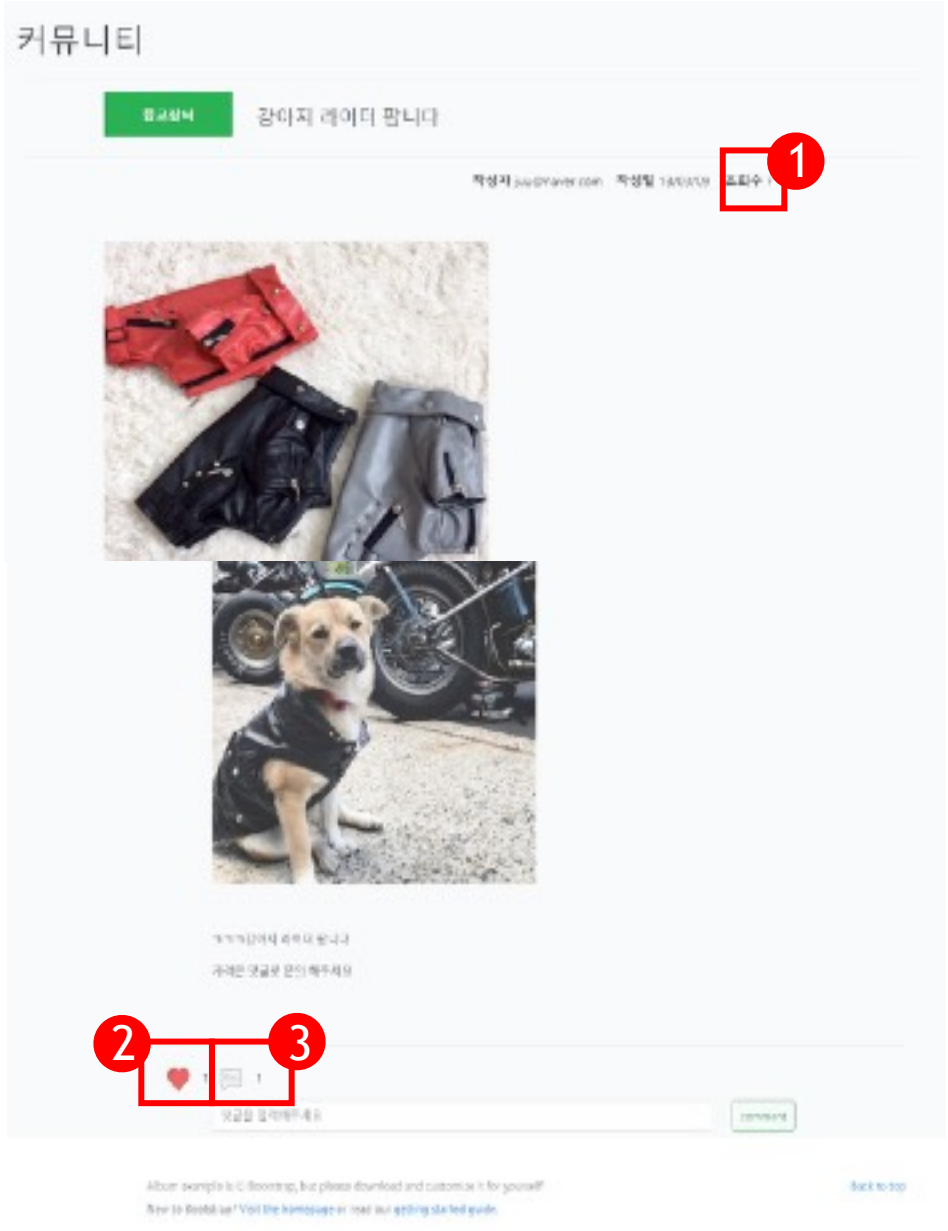
DengDengE 개발 기능(상세)

3. 디테일 페이지(커뮤니티)



Controller

```
String Sseq = req.getParameter("seq");
int seq = Integer.parseInt(Sseq);

HttpSession session = req.getSession();

User current_user = (User) session.getAttribute("current_user");

if (Delegator.checkSession(req, resp)) {

   CommuBbsService comService = CommuBbsService.getInstance();
   CommuBbsCommentService commentService = CommuBbsCommentService.getInstance();
   List<CommuBbsComment> comments = commentService.getAllComments(seq); 4
   comService.readCount(seq); 1
   CommuBbsDto comdto = comService.getCommu(seq);
   boolean isLiked = comService.Prevent_duplication(current_user.getSeq(), seq); 2
   req.setAttribute("comments", comments);
   req.setAttribute("comdto", comdto);
   req.setAttribute("like_count", comService.getLikeCount(seq)); 3
   req.setAttribute("isLiked", isLiked);
   dispatch("CommuBbsDetail.jsp", req, resp);
```

1. 글 읽을 때 readCount 함수 불러와 조회수 올려준다
2. Session에 저장된 유저 정보와 게시물의 시퀀스 번호를 DB를 통해 불러와 좋아요 눌렀으면 true, 아니면 false를 리턴한다
3. 게시물의 시퀀스 번호를 넣어 좋아요 카운트 리턴한다
4. 댓글을 불러온다.

DengDengE 개발 기능(상세)

3. 디테일 페이지(커뮤니티)

<좋아요 기능>

1

```
$('#btnLike').click(function () {

    $.ajax({
        url:"CommuBbsController",
        data: {command: 'like', seq: ${comdto.seq }, userid: ${current_user.seq }},
        type:"post",
        success : function (data) {

            var result = JSON.parse(data);

            if(result.status == 404){
                $("img#like_img").attr("src", "./img/empty_heart.png");
            } else {
                $("img#like_img").attr("src", "./img/heart.png");
            }

            $('span#like_count').html(result.like_count);
        }
    })
});
```

좋아요 버튼 눌렀을 때 ajax 함수를 통해 좋아요 버튼 상태를 변경하고 카운트 수 증가시켜줍니다.

controller

```
String Sseq = req.getParameter("seq");
int seq = Integer.parseInt(Sseq);
String Suser = req.getParameter("userid");
int user = Integer.parseInt(Suser);
System.out.println("seq " + seq + " userid " + user);

CommuBbsService comService = CommuBbsService.getInstance();

int like_count = 0;
HashMap<String, Integer> status = new HashMap<>();

boolean check = comService.Prevent_duplication(user, seq);

if (check) {
    // 테이블에서 해당 행을 삭제( 추가) 한다.
    comService.likeTB_delete(user, seq);

    // status, like count 를 json으로 전송한다.
    status.put("status", 404);
} else {
    // 테이블에서 해당 행을 삭제( 추가) 한다.
    comService.likeTB_insert(user, seq);

    // status, like count 를 json으로 전송한다.
    status.put("status", 200);
}

// 테이블을 게시글 seq 로 count(*)
like_count = comService.getLikeCount(seq);
status.put("like_count", like_count);
String json = new Gson().toJson(status);

System.out.println(json);
resp.getWriter().write(json);
```

2. HashMap으로 status 보냄 (202 : 좋아요, 404 : 취소)
3. 게시물 클릭한 유저가 좋아요 눌렀나 안눌렀나 확인하는 함수
    - 좋아요 테이블에 (user_seq, bbs_seq) 쌍이 존재하는지 확인
4. Json파일을 Gson으로 변환해 보낸다