# " Hello World!! "

열정가득 새싹 개발자 **김유정** 입니다.

Tel. 010-0000-0000
Email. yjkim7948@naver.com
서울특별시

# CONTENTS


01. 개요 프로젝트 소개, 개발동기, 개발단계

02. 기술 사용 기술, 라이센스

03. 설계 DB설계, url설계

04. 개발 일정설계, 주요기능명세

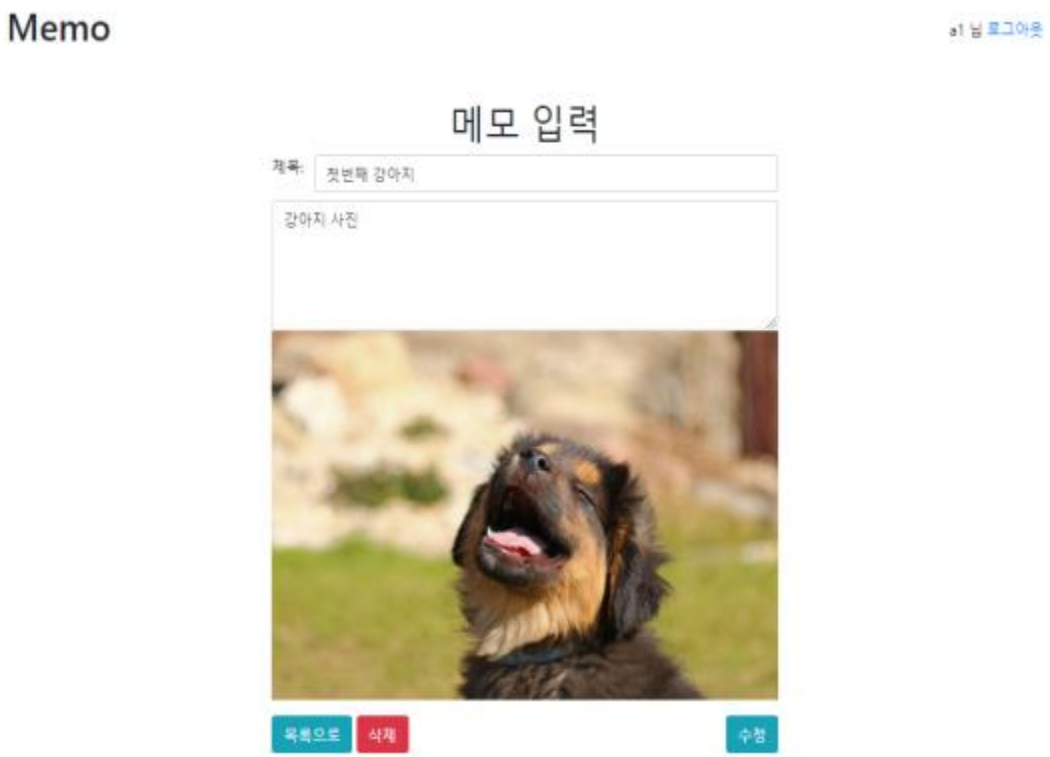
05. 문제해결 문제해결 경험

06. 느낀점 느낀점

# 01. 개요

프로젝트 소개, 개발동기, 개발단계

## 프로젝트 1. "memo"



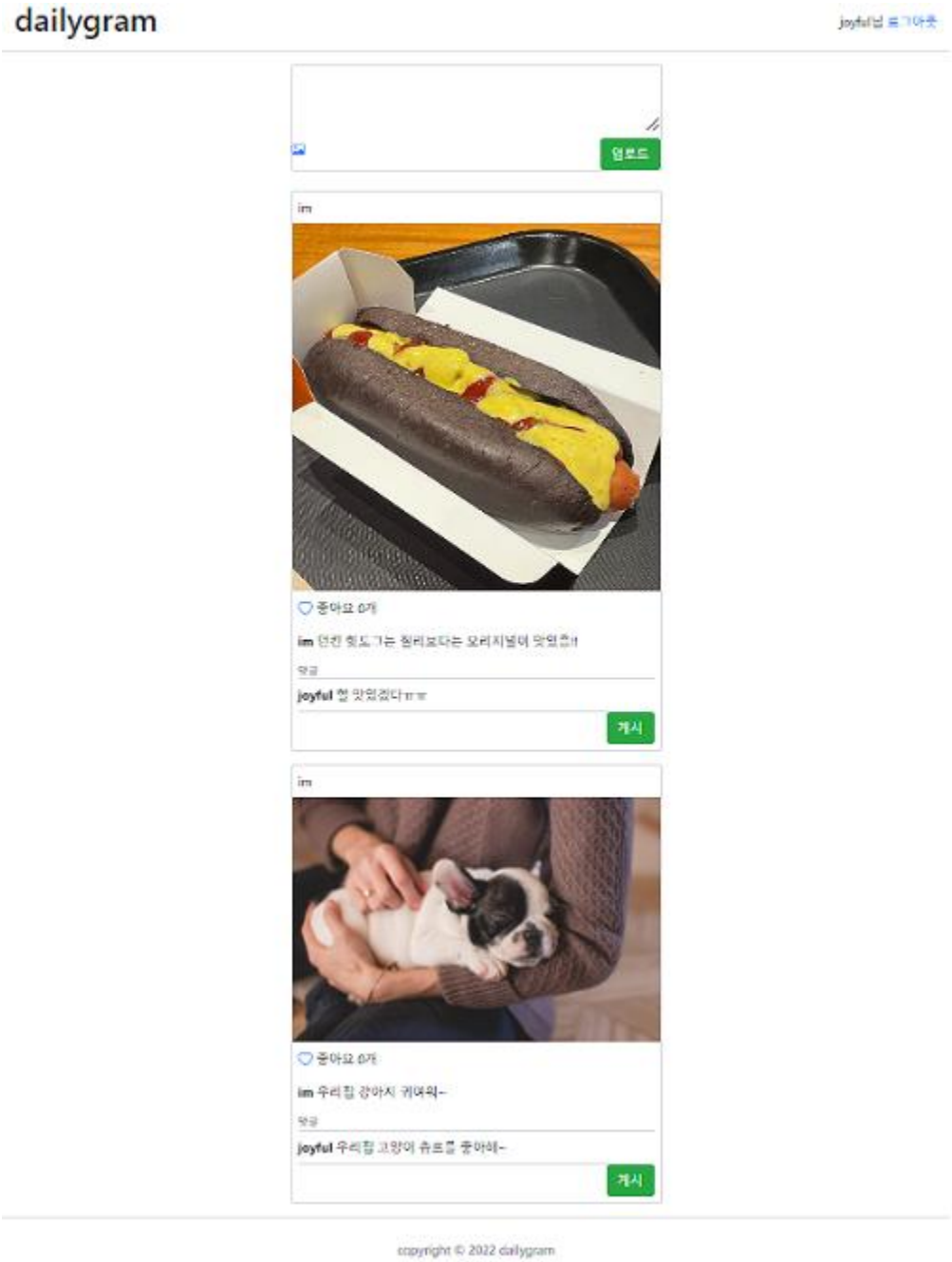

프로젝트 명: Memo

사용자는 로그인 후 서비스를 이용하게 되며, 본인의 글만 보여줍니다.

사용자는 제목, 글 내용, 이미지를 작성 및 저장 할 수 있습니다.

리스트에서 제목을 클릭하면 작성했던 글이 보여지며, 글을 수정할 수 있습니다.

# 프로젝트 2. "Dailygram"




프로젝트 명: Dailygram

사용자는 로그인 후 서비스를 이용하게 되며, 회원가입한 모든사람의 글을 한눈에 확인할 수 있습니다.

사용자는 글 내용, 이미지를 작성 및 업로드할 수 있습니다.

또한 본인이 업로드한 게시글을 삭제할 수 있습니다.
(단, 본인이 업로드한 글만 가능)

모든사용자는 원하는 글에 댓글과 좋아요를 표시할 수 있습니다.

# 프로젝트 3. "PuppyPlay"

## - 프로젝트소개

### PuppyPlay란?

애견유치원 이용 고객을 대상으로 제공되는 서비스로,
월별 일정 공지, 하루 일과 안내, 공지사항, Q&A 작성 및 답글 기능을 제공합니다.

## - 개발동기

SNS를 하면서 강아지유치원의 존재를 알게되었습니다.
미래에 강아지를 키우면서 회사를 다닐경우, 강아지를 혼자 집에 두고 싶지 않았습니다.
이에 강아지 유치원을 알아보았습니다.

현재 서비스되고 있는 강아지 유치원 홈페이지들은 오직 홍보 및 입학용으로 사용되고 있었습니다.
이에, 회원들에게 다양한 서비스를  제공할 수 있는 프라이빗한 홈페이지를 개발하고 싶어, puppyplay 프로젝트를 진행하게 되었습니다.

# - 개발 단계

개발 단계는 아래와 같이 진행하였습니다.

# 02. 기술

사용기술, 라이센스

[라이센스]

Tomcat Apache License 2.0

Mysql GPLv2 or proprietary

Spring framework Apache License 2.0

Mybatis Apache License 2.0

Bootstrap MIT License

jQuery MIT License

FullCalendar MIT License

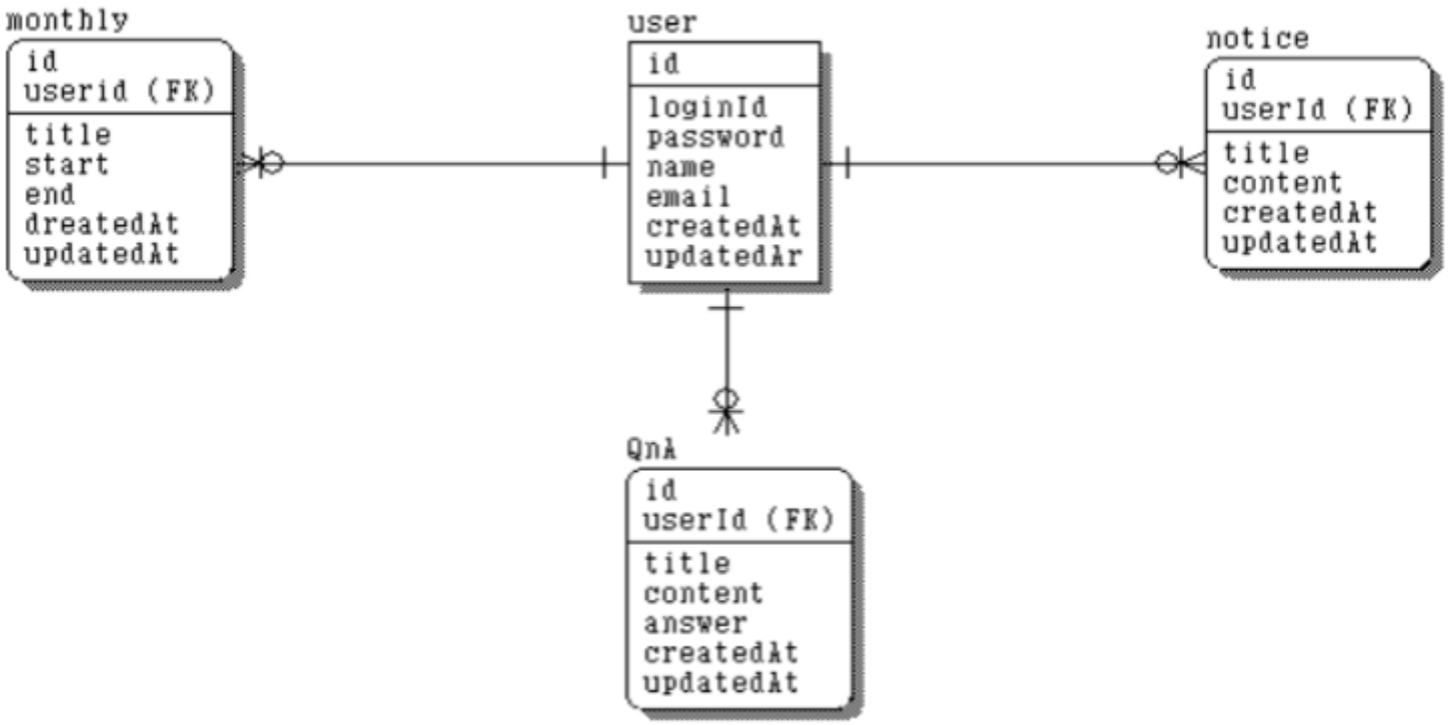
# 03. 설계

DB설계, URL설계

## DB설계(ERD,표)

### - DB설계(ERD)

content, answer,title을 저장하는 테이블에서 사용자 정보가 필요하므로 user테이블의 id 컬럼을 사용할 수 있도록, 테이블 간의 관계들을 만들었습니다.



## - DB 설계

### user

사용자 정보 저장 테이블

| 컬럼이름 | 타입 | null여부 | auto_increment | 설명 |
|---|---|---|---|---|
| id | int | N | Y | primary key |
| loginId | varchar(16) | N | N | 로그인아이디 |
| password | varchar(64) | N | N | 비밀번호 |
| name | varchar(16) | N | N | 사용자이름 |
| email | text | N | N | 이메일 |
| createdAt | timestamp | N | N | 생성시간 |
| updatedAt | timestamp | N | N | 수정시간 |

### monthly

이달의 일정 작성 테이블

| 컬럼이름 | 타입 | null여부 | auto_increment | 설명 |
|---|---|---|---|---|
| id | int | N | Y | primary key |
| userId | int | N | N | 이달의 일정 작성자 user 테이블 (pk) |
| title | varchar(32) | N | N | 제목 |
| start | timestamp | N | N | 일정 시작일 |
| end | timestamp | N | N | 일정 마감일 |
| createdAt | timestamp | N | N | 생성시간 |
| updatedAt | timestamp | N | N | 수정시간 |

### notice

공지사항 작성 테이블

| 컬럼이름 | 타입 | null여부 | auto_increment | 설명 |
|---|---|---|---|---|
| id | int | N | Y | primary key |
| userId | int | N | N | 공지사항 작성자 user 테이블 (pk) -관리자 |
| title | varchar(16) | N | N | 제목 |
| content | text | N | N | 공지 글 내용 |
| createdAt | timestamp | N | N | 생성시간 |
| updatedAt | timestamp | N | N | 수정시간 |

### Q&A

Q&A 작성 테이블

| 컬럼이름 | 타입 | null여부 | auto_increment | 설명 |
|---|---|---|---|---|
| id | int | N | Y | primary key |
| userId | int | N | N | Q&A 작성자 user 테이블 (pk) |
| title | varchar(16) | N | N | 제목 |
| content | text | N | N | Q&A 글 내용 |
| answer | text | Y | N | Q&A 답변 글 내용 |
| createdAt | timestamp | N | N | 생성시간 |
| updatedAt | timestamp | N | N | 수정시간 |

# URL 설계(View,API)

효율적인 url 설계를 위해, VIEW URL 과 API URL을 분류하여 설계하였습니다.

VIEW URL: 사용자에게 화면을 보여주기 위해 URL  생성
API URL:  기능수행을 위한 api를 수행하는데 있어, 필요한 method, requestParamter,response 데이터를 정의

## - View URL

### View URL

| 제목 | url | parameter | 설명 |
|---|---|---|---|
| 로그인 | /user/signin/view | | 로그인 화면 |
| 회원가입 | /user/signup/view | | 회원가입 화면 |
| 메인화면 | /post/main/view | | 기본 페이지 화면<br>버튼을 클릭하여, 원하는 기능으로 이동할 수 있다. |
| 이달의 일정 입력 | /post/monthly/create/view | | 달 별 일정을 기록한다.<br>관리자 아이디로만 접근 가능. |
| 오늘의 일정 | /post/today/view | | 하루 단위, 일정을 고지한다. ( 변동 사항 x) |
| 공지사항 입력 | /post/notice/create/view | | 공지사항 입력화면<br>관리자 아이디로만 접근 가능. |
| 공지사항 리스트 | /post/notice/list/view | | 공지사항 목록을 보여준다. |
| 공지사항 보기 | /post/notice/detail/view | id: 공지사항 글에 대한 id | 공지사항의 상세내용을 보여준다.<br>id 파라미터를 통해서 공지사항을 선택하여 보여준다.<br>공지사항 목록에서 선택하여 접근한다. |
| Q&A 입력 | /post/qna/create/view | | Q&A 글 작성 화면 |
| Q&A 댓글 입력 | /post/qna/reply/create/view | id: Q&A 글에 대한 id | Q&A 댓글 작성 화면 ( 관리자만 접근 가능)<br>id 파라미터를 통해서 Q&A글을 선택하여 보여준다. |
| Q&A  리스트 | /post/qna/list/view | | Q&A 화면<br>회원과, 관리자가 작성한 글목록 을 보여준다. |
| Q&A 보기 | /post/qna/detail/view | id: Q&A 글에 대한 id | Q&A 글의 상세내용을 보여준다.<br>id 파라미터를 통해서 Q&A글을 선택하여 보여준다.<br>Q&A 목록에서 선택하여 접근한다. |

## - API URL

| API URL | | | |
|---|---|---|---|
| | | | |
| **1) 로그인** | | | |
| 아이디와 패스워드를 통해서 로그인 기능 수행 | | | |
| url: /user/signin | | | |
| method: post | | | |
| parameter | | | |
| 이름 | 타입 | Null여부 | 설명 |
| loginId | String | N | 로그인 ID |
| password | String | N | 로그인 비밀번호 |
| | | | |
| 응답값(json) | | | |
| 성공시 | {"result" : "success"} | | |
| 실패시 | {"result" : "fail"} | | |
| | | | |
| | | | |
| **2) 회원가입** | | | |
| 가입 정보를 입력받고, 정보를 저장한다. | | | |
| url: /user/signup | | | |
| method: post | | | |
| parameter | | | |
| 이름 | 타입 | Null여부 | 설명 |
| loginId | String | N | 로그인ID |
| password | String | N | 비밀번호 |
| name | String | N | 사용자 이름 |
| email | String | N | 사용자 이메일 |
| | | | |
| 응답값(json) | | | |
| 성공시 | {"result" : "success"} | | |
| 실패시 | {"result" : "fail"} | | |
| | | | |
| | | | |
| **3) 로그아웃** | | | |
| 로그인된 사용자를 로그아웃 시킨다. | | | |
| url: /user/signout | | | |
| method: get | | | |

## 4) 이달의 일정 입력

관리자가 작성한 일정을 입력한다.

url: /post/monthly/create

method: post

parameter

| 이름 | 타입 | Null여부 | 설명 |
|---|---|---|---|
| title | String | N | 제목 |
| start | Date | N | 시작 일 |
| end | Date | N | 종료 일 |

응답값(json)

| | |
|---|---|
| 성공지 | {"result" : "success"} |
| 실패시 | {"result" : "fail"} |

## 5) 이달의 일정 리스트

관리자가 작성한 일정을 보여준다.

url: /post/monthly/list

method: get

## 6) 이달의 일정 삭제

관리자가 작성한 일정을 보여준다.

url: /post/monthly/delete

method: post

## 7) 공지사항 입력

관리자가 작성한 공지사항을 저장한다.

url: /post/notice/create

method: post

parameter

| 이름 | 타입 | Null여부 | 설명 |
|---|---|---|---|
| title | String | N | 제목 |
| content | String | N | 글 내용 |

응답값(json)

| | |
|---|---|
| 성공지 | {"result" : "success"} |
| 실패시 | {"result" : "fail"} |

## 8) 공지사항 수정

관리자가 작성한 공지사항을 저장한다.

url: /post/notice/update

method: post

parameter

| 이름 | 타입 | Null여부 | 설명 |
|---|---|---|---|
| title | String | N | 제목 |
| content | String | N | 글 내용 |

| 응답값(json) | |
|---|---|
| 성공지 | {"result" : "success"} |
| 실패시 | {"result" : "fail"} |

## 9) Q&A 입력

회원이 작성한 Q&A을 저장한다.

url: /post/qna/create

method: post

parameter

| 이름 | 타입 | Null여부 | 설명 |
|---|---|---|---|
| title | String | N | 제목 |
| content | String | N | 글 내용<br>관리자 아이디 일 경우, 답변작성 버튼 활성화 |

| 응답값(json) | |
|---|---|
| 성공지 | {"result" : "success"} |
| 실패시 | {"result" : "fail"} |

## 10) Q&A 답글 입력

관리자가 작성한 Q&A 답글을 저장한다.

url: /post/qna/reply

method: post

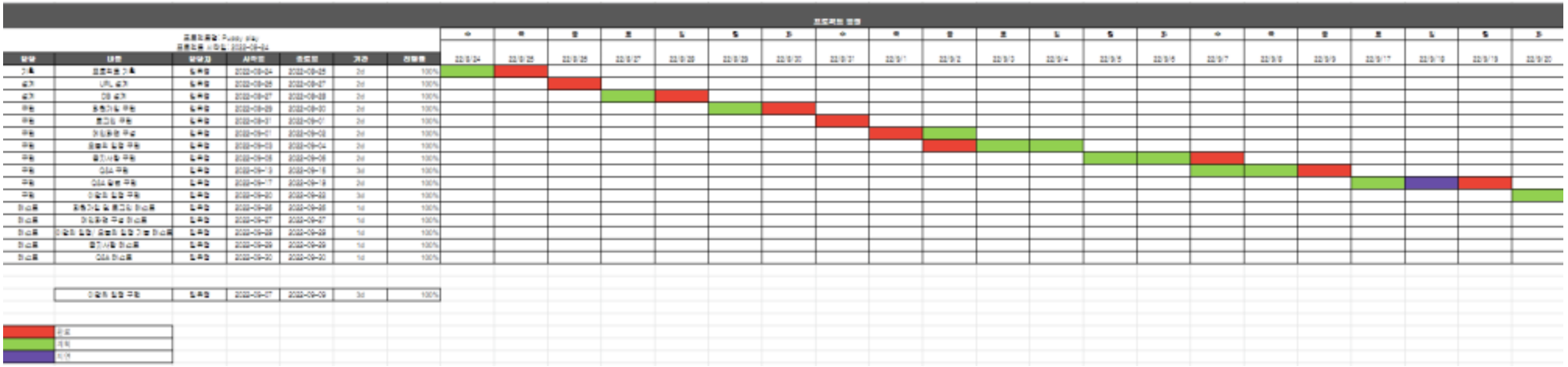
parameter

| 이름 | 타입 | Null여부 | 설명 |
|---|---|---|---|
| postId | int | N | 수정할 메모의 post테이블의 ID (pk) |
| title | String | N | 제목 |
| content | String | N | 답변 글 내용 |

| 응답값(json) | |
|---|---|
| 성공지 | {"result" : "success"} |
| 실패시 | {"result" : "fail"} |

# 04. 개발

## 일정설계, 주요기능명세

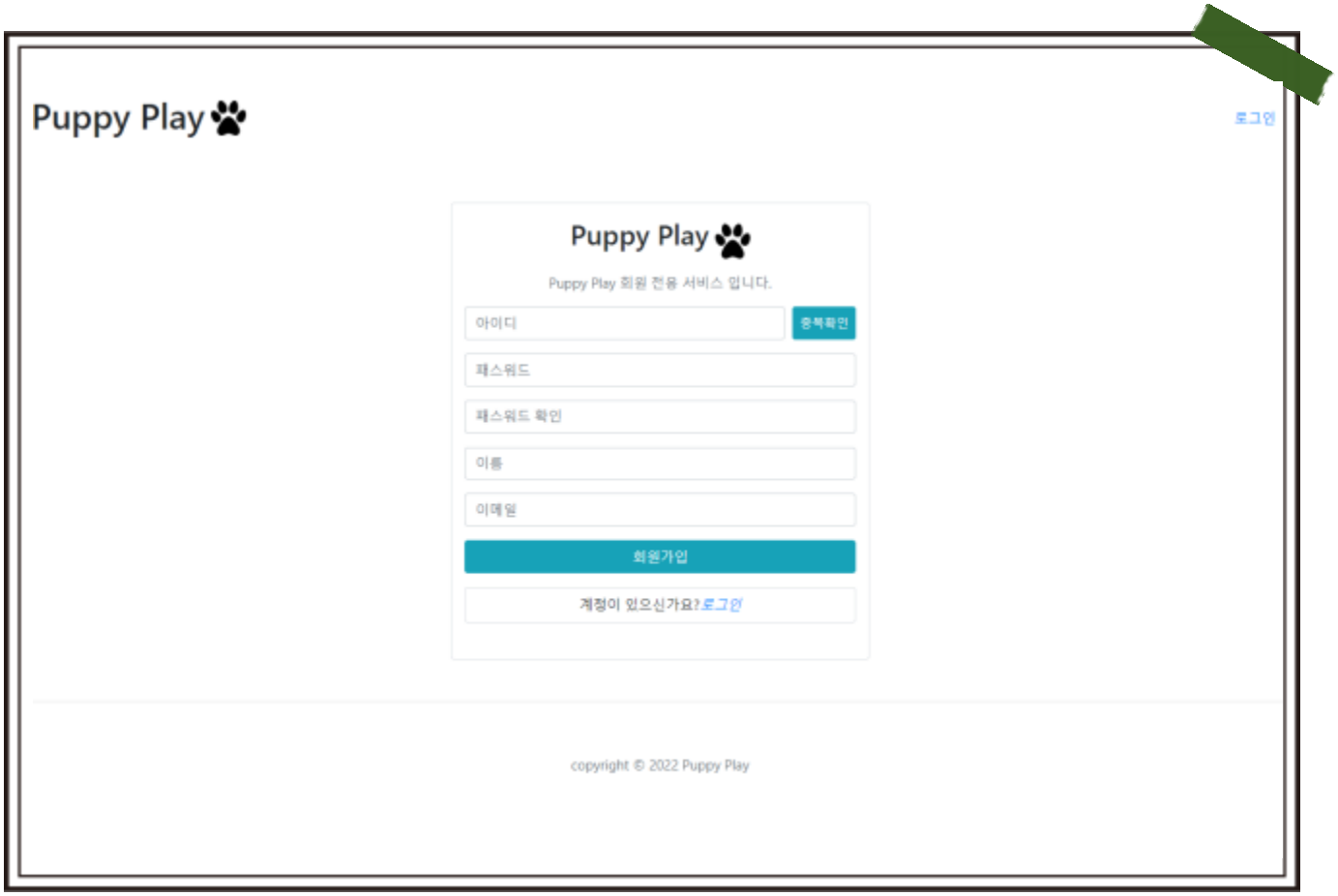


## - 주요기능명세

# 1. 회원가입, 로그인

**"사용자로부터 회원가입에 필요한 정보를 입력받아 db에 저장 및 조회한다."**

사용자로부터 정보를 입력받아, 쿼리를 수행한다.
회원가입의 경우 insert쿼리를 수행하며 로그인은 select 쿼리를 수행합니다.



[아이디 중복 확인 기능]

RestContorller에서 전달받은 아이디가 테이블에 존재하는 지 확인 한 후,  존재할 경우 {"result" : "is_duplicate"} 를 전달합니다.

[세션 기능]

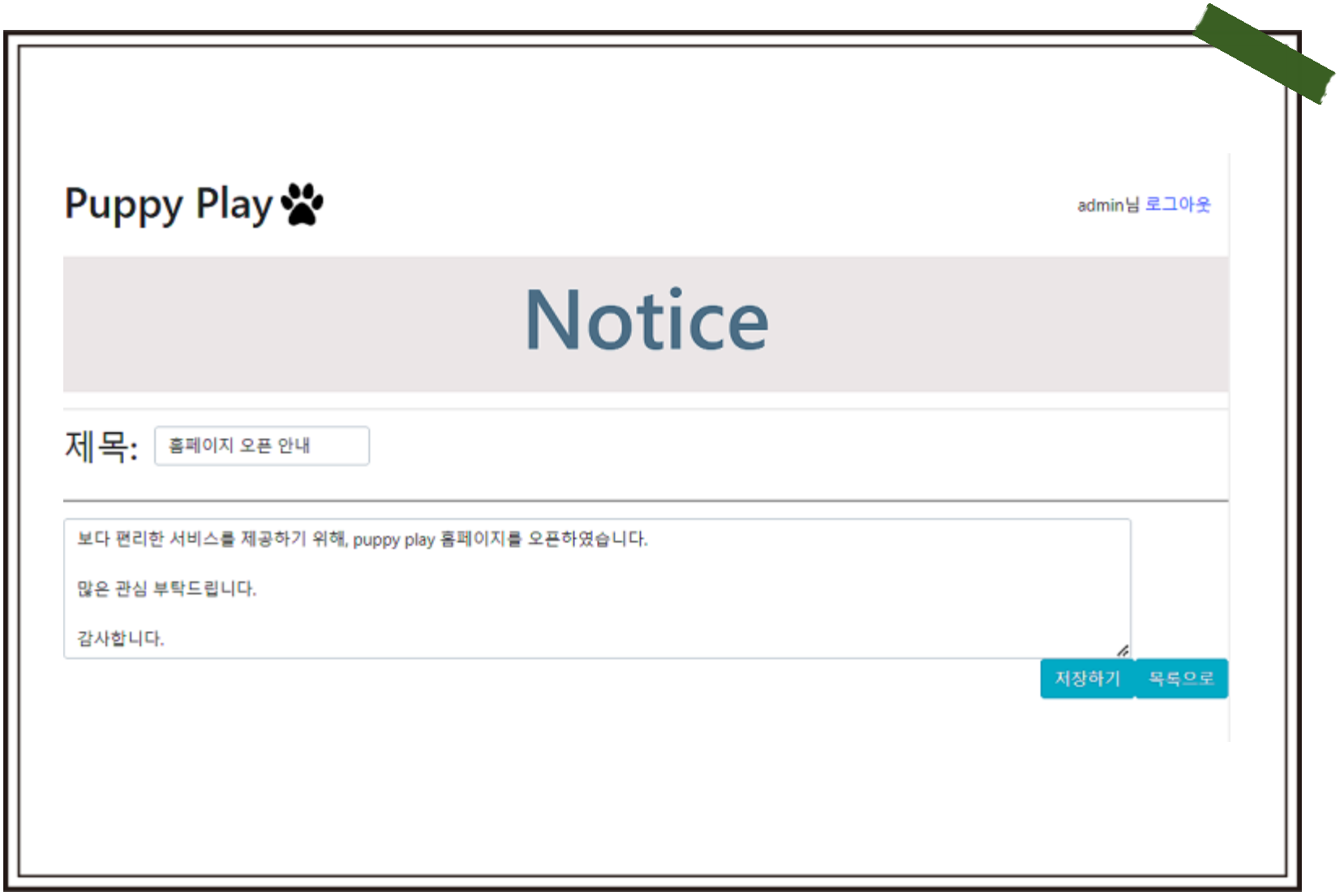
세션은 쿠키와 다르게 사용자의 정보가 서버에 저장되기 때문에,
session을 통해 사용자의 정보를 서버에 저장하고, 원하는 곳에서 값을 사용할 수 있습니다.
로그인 성공시, HttpServletRequest 객체의 getSession() 메소드를 이용하여 session을 생성하였습니다.
setAttribute()메소드를 이용해, "userId"에 로그인한 사용자의 id 값을,  "userLoginId"에 로그인 사용자의 로그인id 값을 저장합니다.

[비밀번호 암호화 기능]

보안성을 위해 비밀번호를 암호화하여 저장했습니다.
암호화 방식으로는 MD5 알고리즘을 사용하였습니다.
password를 입력하면, 해당 문자열을 byte화 하여 저장하고,  저장한 결과를 16진수 문자열로 변환하였습니다.
사용자가 비밀번호를 입력하면, 암호화된 비밀번호와 비교하여 일치하는 값을 찾으면 {"result":"success"}를 전달합니다.

# 2. 공지사항

"글을 작성 및 수정할 수 있다."
"글목록에서 원하는 제목을 클릭하여 내용을 확인할 수 있다. "



[글 목록]

글 목록의 구성요소로는 [글 번호, 글 제목, 글쓴이, 작성날짜]가 있습니다.
원하는 제목을 클릭하면, 상세 페이지로 이동할 수 있게 만들었습니다.
페이지 하단에, 관리자만 보이는 작성하기 버튼을 생성하여, 관리자만 작성화면으로 이동할 수 있게 만들었습니다.
jsp에서 사용할 수 있돋록 model을 사용했으며, db조회 결과 얻어온 데이터를 List로 저장했습니다.
<table>태그를 사용하여 테이블을 구성하였으며
<tbody>부분에서 jstl core라이브러리를 이용해, forEach 반복문을 사용했습니다.
<tr>안의 <td> 값들은 model의 List 값을 가져와 사용했습니다.

[글 작성]

유효성 검사를 통해 잘못된 결과가 넘어올 경우 {"result":false"}를, 이상이 없을 시, ajax를 호출하였습니다.

호출 성공 시, 목록 페이지로 이동하며
오류 발생 시, alert()를 통해 오류 메세지를 나타냈습니다.

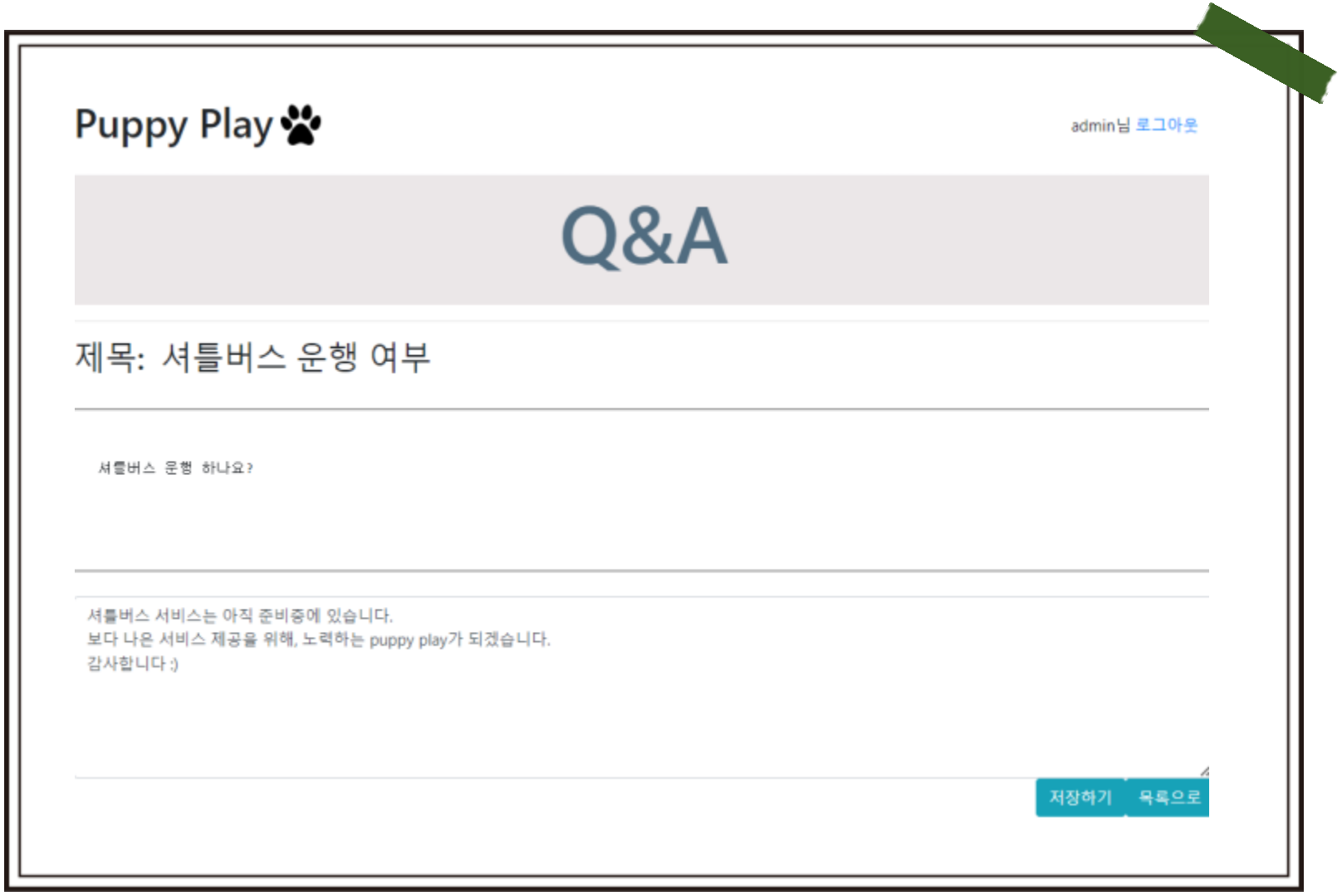
정확한 구분을 위해 data 전달 성공 여부에 따라 alert 메세지를 세부적으로 분류하였습니다.
data 전송 success->오류 발생 : alert("실패")
data 전송 error->alert("에러")

[글 상세 페이지]

회원은 작성된 글 내용을 확인 할 수 있으며, 관리자는 글 내용을 수정할 수 있는 api를 만들었습니다.
jstl core라이브러리를 이용해, choose 태그를 사용했습니다.
userLoginId가 'amin'일 경우, <input>태그를 통해 값을 입력(수정)할 수 있도록 하였고
그 외에는 작성된 값을 불러왔습니다.
글 수정은 ajax를 통해 수행하였습니다.

# 3.Q&A

"기본적인 글 작성 기능을 이용할 수 있다."
"1:1 답변 기능을 이용할 수 있다."



[글작성, 글목록]

공지사항과 흐름이 동일합니다.

[답변기능]

원하는 글 제목을 클릭하면 해당하는 글이 보여지며,
관리자는 textarea를 통해 답변을 작성할 수 있습니다.
jstl core라이브러리를 사용하여 사용자 로그인 아이디가 'admin'일 경우에만 textarea가 보여지도록 만들었습니다..
유효성 검사를 통해 이상이 없을 경우 ajax를 호출했습니다..

[**어려웠던 점]

Q&A의 페이지구조를 설계하는것이 어려웠습니다.
처음에 create page에 글 작성, 답글작성 폼을 모두 배치하여 진행했었습니다.
"회원이 게시글 작성 및 저장 -> 관리자가 글 목록에서 원하는 글 선택-> detail 페이지에서 사전에 작성된 회원의 글을 보며 답변 작성"의 흐름으로 설계하고 작업을 진행했습니다.

답변기능을 구현하다, 눈앞이 깜깜해지기 시작했었습니다.
여러가지 문제가 발생했고 사고가 정지되는 일이 발생했습니다.
그러다 문득 해결 방법으로 두가지가 떠올랐습니다.

떠올랐던 해결방법 두가지는 다음과 같습니다.

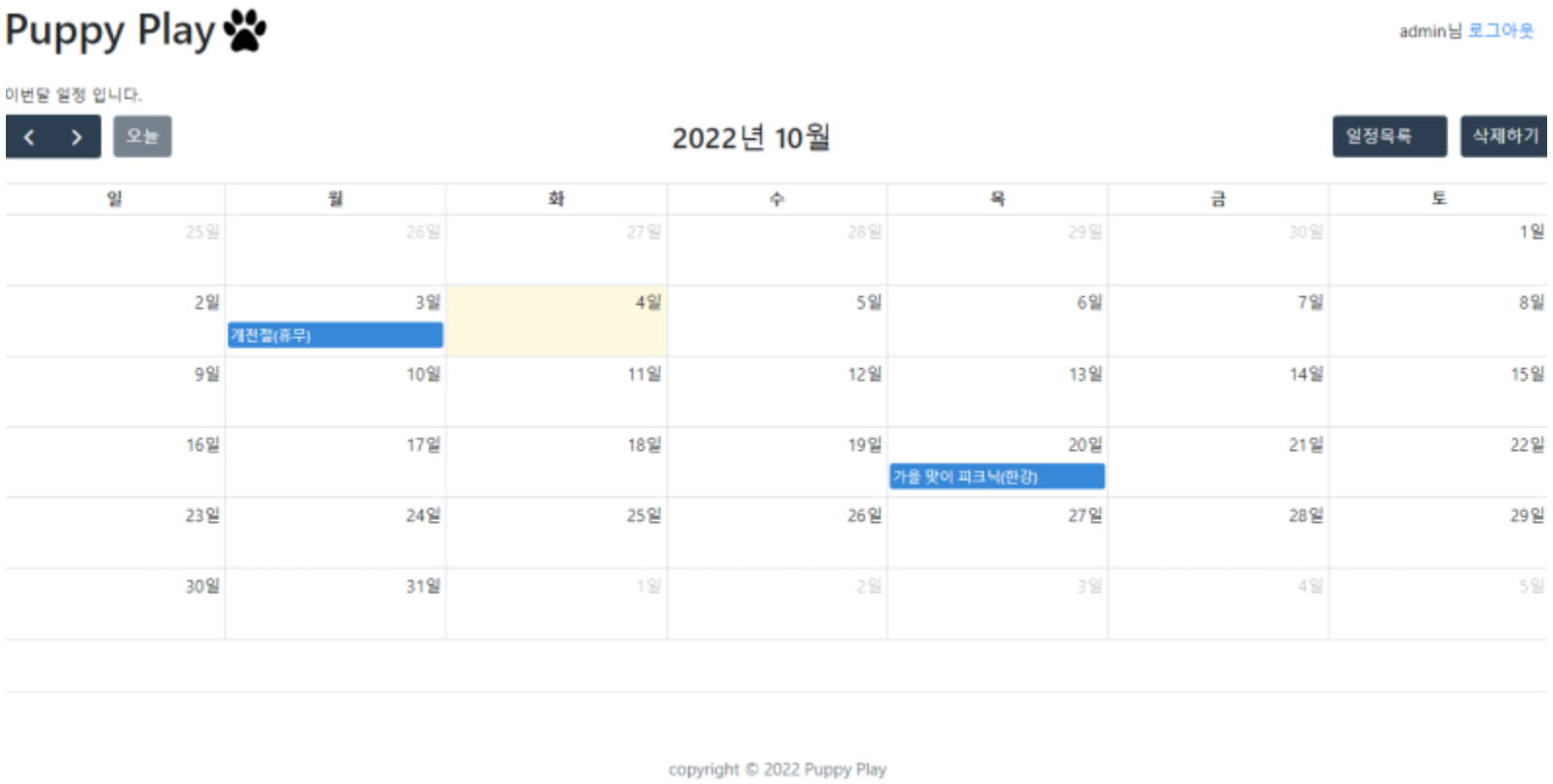
첫번째, 답변작성 폼을 create에 만들고 사용하지 않는 방법(추후, update쿼리 사용)
두번째, detail page에서 회원이 작성한 글을 불러오고, 그 밑에 답변 구현 form을 넣어서 update를 하는 방법
중에 고민을 했습니다.
첫번째 방법은 폼을 만들고 사용하지 않을 경우 500 ERROR가 발생할 것이기에 불가능할 거 같았습니다.
두번째 방법은 현재 같은 'qna' table에 들어있는 answer 컬럼과 content 컬럼을 분리해야했기에 아예 table 설계부터 다시해야 했습니다.

그러다 문득, db 설계시 answer를 null 허용으로 하면 해결이 되겠다는 아이디어가 떠올랐고, db에서 answer을 null 허용으로 변경하고,
첫번째방법으로 진행하여 문제를 해결하였습니다.

# 4. 이달의 일정

## "관리자가 일정을 등록하여 사용자들에게 보여준다."

오픈소스 라이브러리인 Fullcalendar를 사용하였습니다.
일정 생성, 일정 출력, 일정 삭제 기능을 구현했습니다.
(단, 관리자 아이디일 경우에만 이벤트 생성 가능)



[일정 생성]

생성한 Fullcalendar 객체에  "selectable: true" 를 설정하여 드래그로 이벤트를 생성할 수 있게 만들었습니다.
사용자가 이벤트를 드래그하면 선택된 날짜의 정보들이 전달된다.
일정에는 [title,start(시작날짜),end(끝나는날짜)] 값들이 들어있으며, null 값이 아닐 경우, ajax를 통해 "post/monthly/create"를 실행하여 일정을 생성합니다.
관리자만 일정 생성이 가능할 수 있게, RestController에서 if문을 통해 if(userLoginId == "admin")일 경우에만 count 값을BO로 부터 가져올 수 있게 만들었습니다.

[일정출력]

생성한 일정(events)을 배열 형식으로 저장하며, 배열 안의 각 데이터는 map을 이용하여 값을 넣어주었습니다.
이에, 데이터 타입은 List<Map<String,String>>으로 선언하였습니다.
데이터 처리는 BO에서 진행하였으며, DAO에서 select해온 데이터들을 List로 저장하고,
향상된 for문을 통해 각 각 데이터를 얻어오고, Map을 통해 값을 넣어주었습니다.

[일정삭제]

생성한 Fullcalendar 객체에  "droppable: true" 를 설정하여 삭제 이벤트를 등록한였습니다.
"customButtoms"를 등록하여 삭제하기 버튼을 추가하였으며, click 이벤트를 걸어,
click시 calendar.removeAllEvents()과 ajax 호출이 진행되도록 하였습니다.
ajax호출 시, "post/monthly/delete"가 진행되어 저장되어있던 모든 데이터가 사라집니다.

[**어려웠던점]

오픈소스 라이브러리를 써보고자 구글에 검색을 해보았지만, 제겐 아직 어려운 코드들이 대부분이였습니다.
단순 입력 코드는 쉽게 찾았지만, db에 저장하는 코드 중 제가 참조할만한 코드를 찾는게 제일 어려웠습니다.
또한 찾았다해도 원하지 않는 기능들도 같이 섞여 있어, 정확하게 분별해내는 과정이 어려웠습니다.

가장 간단한 틀로 만든 후, 본격적으로 insert를 시작했을 때, 데이터 타입을 찾는게 잘 떠올르지 않아 티스토리를 참고해 List 형식으로 만들어 해결했습니다.
그 후, 데이터 타입을 List<Map<>> 로 잡는것까지는 매우 어렵진 않았습니다.
다만 이를 반복하기 위한 "향상된 for문"을 떠올리는것이 어려웠습니다.
또한 map을 통해 객체들을 채워준 후, 원래형태인 List<Map<>>에 add하여 return 하는 과정이 가장 복잡했습니다.

# 05. 문제해결

[Q&A 문제해결]

answer기능 구현에 'update'쿼리를 사용해보기로 했습니다.
db 설계시, answer값을 null 허용으로 만들어, 처음 게시물 입력시 null 값으로 전달이 될 수 있도록 설계하였습니다.

그러나, 아직 개발에 서툴렀던 저는 null 허용으로 만들면 아무런 입력을 안해도 전달이 될 줄 알았기에,
answer를 create의 parameter로 넣되, 아무 입력도 하지 않았습니다.
결국 예상했던대로 "500 ERROR" 가 발생했습니다.
마침내 다음날 위 문제를 해결했으나, 값이  null 값이 아닌 빈 문자열로 전달되는 문제가 생겼습니다.

해당 문제로 인해, answer를 update 하기만 하면, 기존에 작성되었던 answer의 값도 모두 빈 문자열로 변경되는 문제가 발생했습니다.

어떤 부분이 문제일지 감이 안오는 저는, 삼일 동안 코드를 천천히 보며 도대체 무엇이 문제일 지, 디버깅을 시작했습니다.

디버깅 결과, 어디선가 빈 문자열이 전달되어 작성된 텍스트를 빈값으로 update하고 있음을 알게되었습니다.
이는 update 수행쪽 문제기 때문에 Mapper ~ DAO 를 확인 한 후, ajax 쪽을 살펴보았습니다.
문제는 if(answer == "")를 if(answer = "") 로 = 하나가 빠져있었습니다. 즉, 빈 문자열을 대입하는 오타 문제 였습니다.
이를 수정하고 나니, 원만하게 update가 진행되었습니다.

[이달의 일정]

Fullcalendar에서 관리자만 일정을 작성할 수 있게 하는 과정에서, if문을 실행하여, userLoginId = 'admin' 일 경우에만 count과정이 진행되게 하고싶었습니다.

그러나, 기존 방식대로 if(조건){int count = monthlyBO.addMonthly}Map<String,String> retult = new HashMap<>();를 실행하니 if 문 밖에 있는 map이 실행이 안되는 문제가 발생했습니다.

이에, count 변수를 if문 밖에 선언하고 초기화하였더니 정상적으로 실행이 가능하게 됐습니다.

[그외]

학교에서 사용하는 노트북에 mysql을 설치하는데, 3306 포트가 좀비처럼 알아서 사용되고 있었습니다.
분명 집에 있는 노트북이랑 똑같은 방법으로 설치하는데, 오류가 나서 사고정지가 왔었습니다.
다시 마음을 다잡고 구글링을 통해 사용중인 3306 port 없애기를 검색하였습니다.
검색 후, cmd를 켜 "netatat -ano | find "3306" 을 실행해, 3306 포트를 사용하고 있는 PID를 찾았습니다.
그리고 "taskkill /F /PID 포트번호"  명령을 실행 해 다시 3306포트를 되찾을 수 있었습니다.

# Thank you

열정가득 새싹 개발자 김유정 이였습니다.

감사합니다 :)