

Yoony **CALENDAR**

By Yoon Gayoung

<https://github.com/Yoon1717/YoonyCalendar.git>



Yoony CALENDAR

Leave your day, leave your moment

- 01 프로젝트 주제
- 02 주요 기능
- 03 프로젝트 핵심 소스
- 04 향후 개선점



● 간편한 사용성을 갖춘 클릭 기반의 웹 서비스 < >

사용자는 감정과 관련된 경험을 아이콘과 함께 간편하게 기록

사용자의 일상적인 감정을 시간에 따라 추적

특히, 사용자가 감정과 관련된 아이콘을 등록할 수 있도록 함으로써,

사용자의 일상적인 감정 변화를 분석하고 시각화

개인의 감정 패턴을 이해하고 관리할 수 있는 도구를 제공

02 주요 기능



The screenshot shows a calendar interface for November 30, 2023, featuring a dark background with a landscape image of two people walking in a field. The top navigation bar includes links for HOME, 로그인 (Login), and 회원가입 (Sign Up). The main calendar area displays a grid from 12am to 7am. A modal window in the bottom right corner contains the text "localhost:8080 내용:" and "아이콘 등록은 로그인 이후 작성 가능합니다" (Icon registration is available after logging in). A blue "확인" (Confirm) button is at the bottom right of the modal. A small tooltip in the top right corner of the calendar area says "파일 선택 선택된 파일 없음" (File selection: No selected file) and "코멘트를 입력하세요" (Enter a comment).

메인 페이지

로그인/회원가입

아이콘 등록의 경우
로그인 후 이용 가능

02 주요 기능



The screenshot shows a login form on a website. At the top, there is a dark header bar with three white buttons: 'HOME', '로그인' (Login), and '회원가입' (Sign Up). Below the header, the word '로그인' is centered in large white text. The main area contains two input fields: the first is labeled '아이디' (ID) and contains the text 'Yoony'; the second is labeled '비밀번호' (Password) and contains several dots. Below these fields is a checkbox labeled '아이디 기억하기' (Remember ID) with an unchecked state. At the bottom right of the form is a yellow '로그인' (Login) button. The background of the page features a dark landscape image of a person walking.

Copyright © Your Website 2023

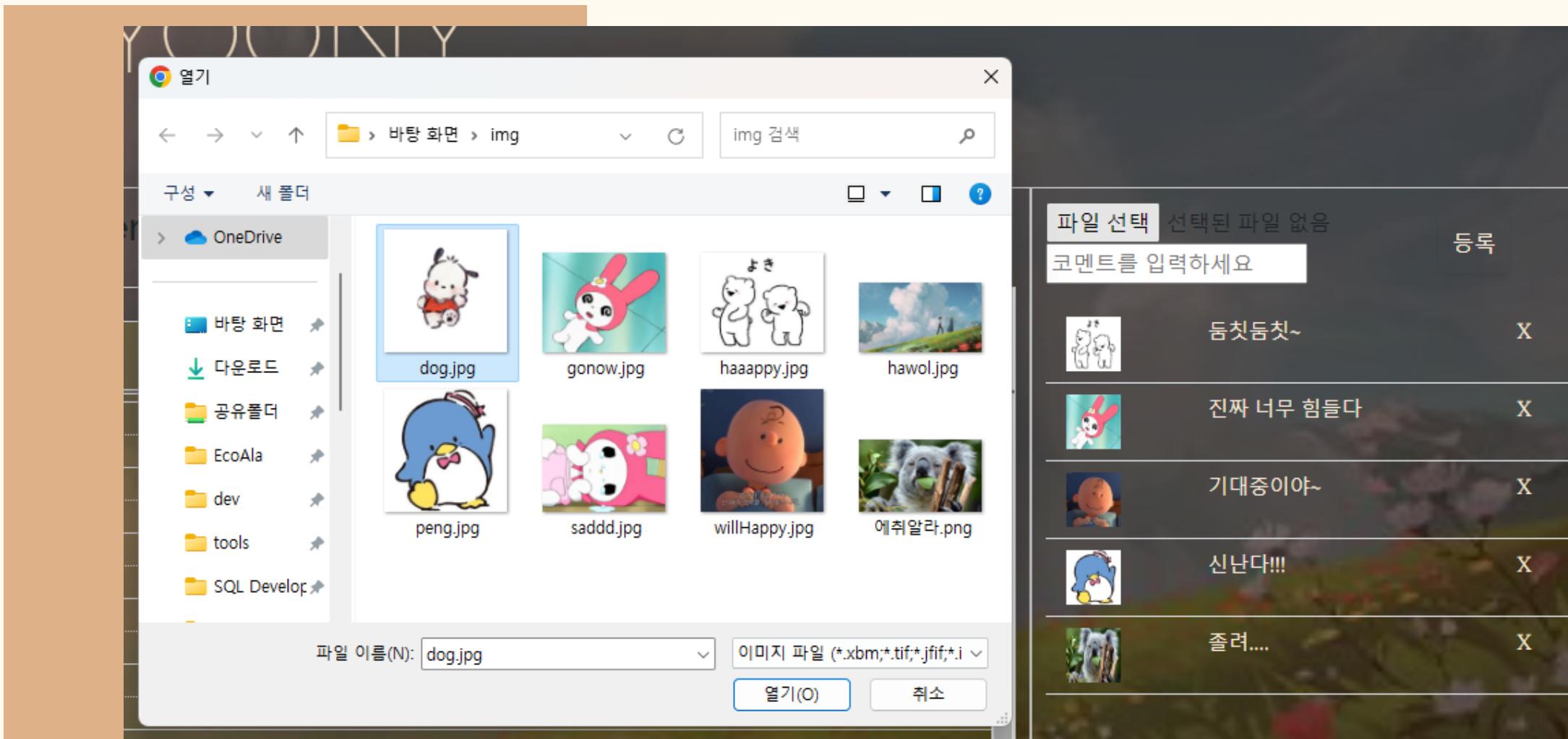
The screenshot shows a sign-up form on a website. At the top, there is a dark header bar with three white buttons: 'HOME', '로그인' (Login), and '회원가입' (Sign Up). Below the header, the word '회원가입' is centered in large white text. The main area contains four input fields: the first is labeled '아이디' (ID) and has '아이디' typed into it; the second is labeled '비밀번호' (Password); the third is labeled '이름' (Name); and the fourth is a yellow '가입하기' (Sign Up) button. The background of the page features a dark landscape image of a person walking.

Copyright © Your Website 2023

로그인/회원가입 페이지

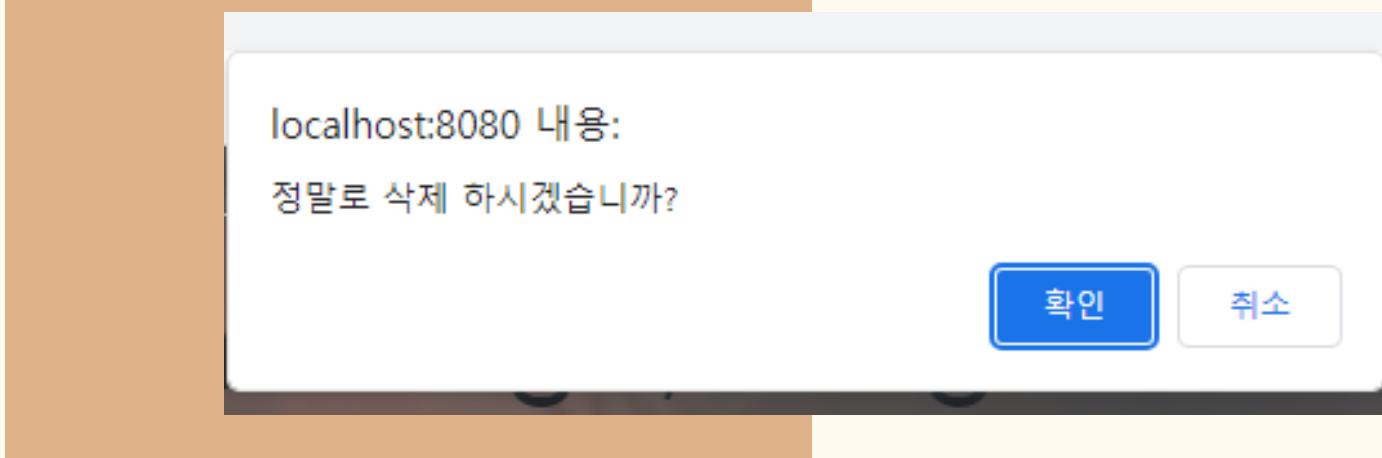
비밀번호 암호화

02 주요 기능



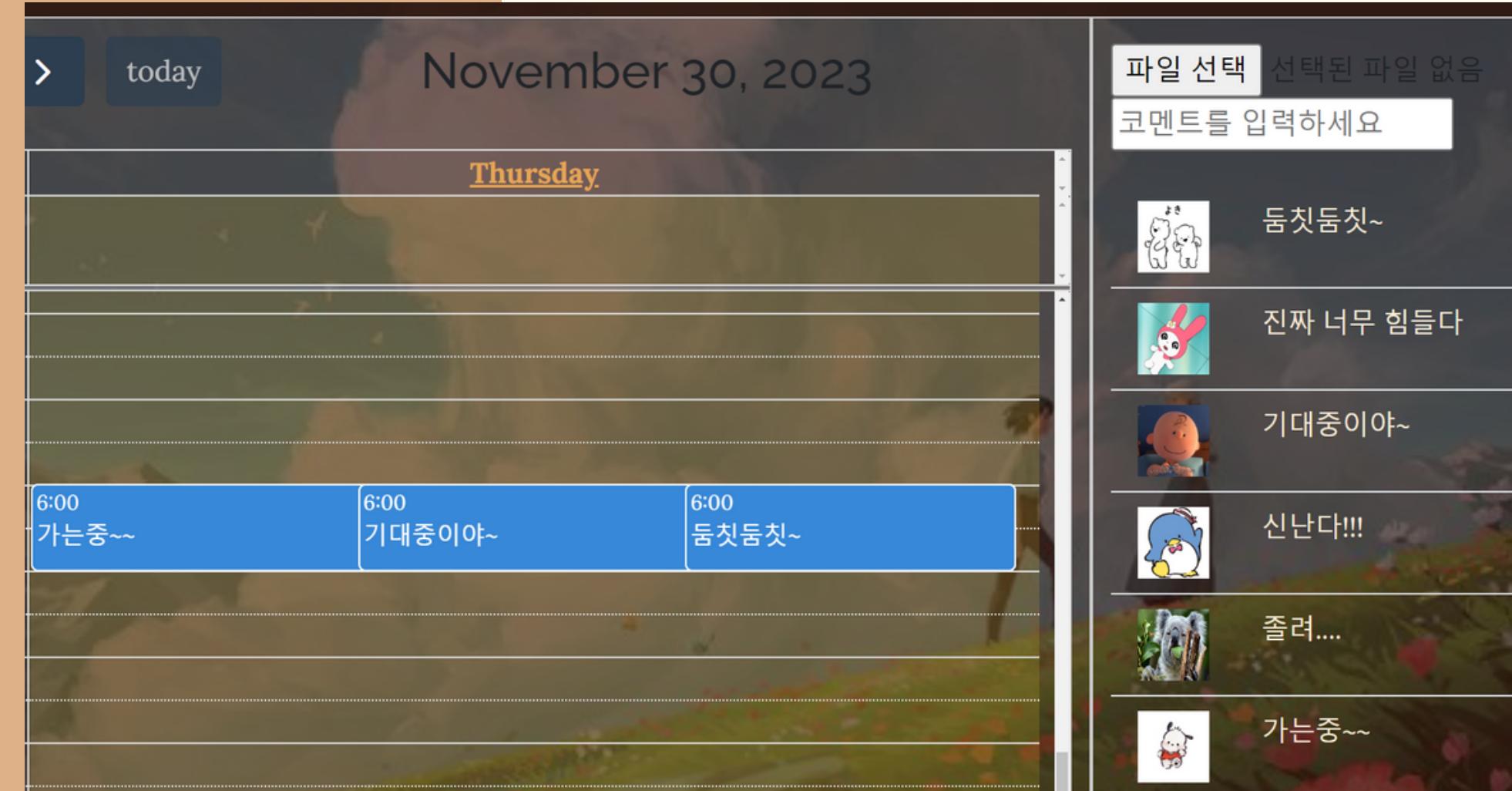
아이콘 등록

회원은 원하는 이미지와
코멘트를 입력하면
아이콘 등록이 가능



필요에 따라 삭제 가능

02 주요 기능

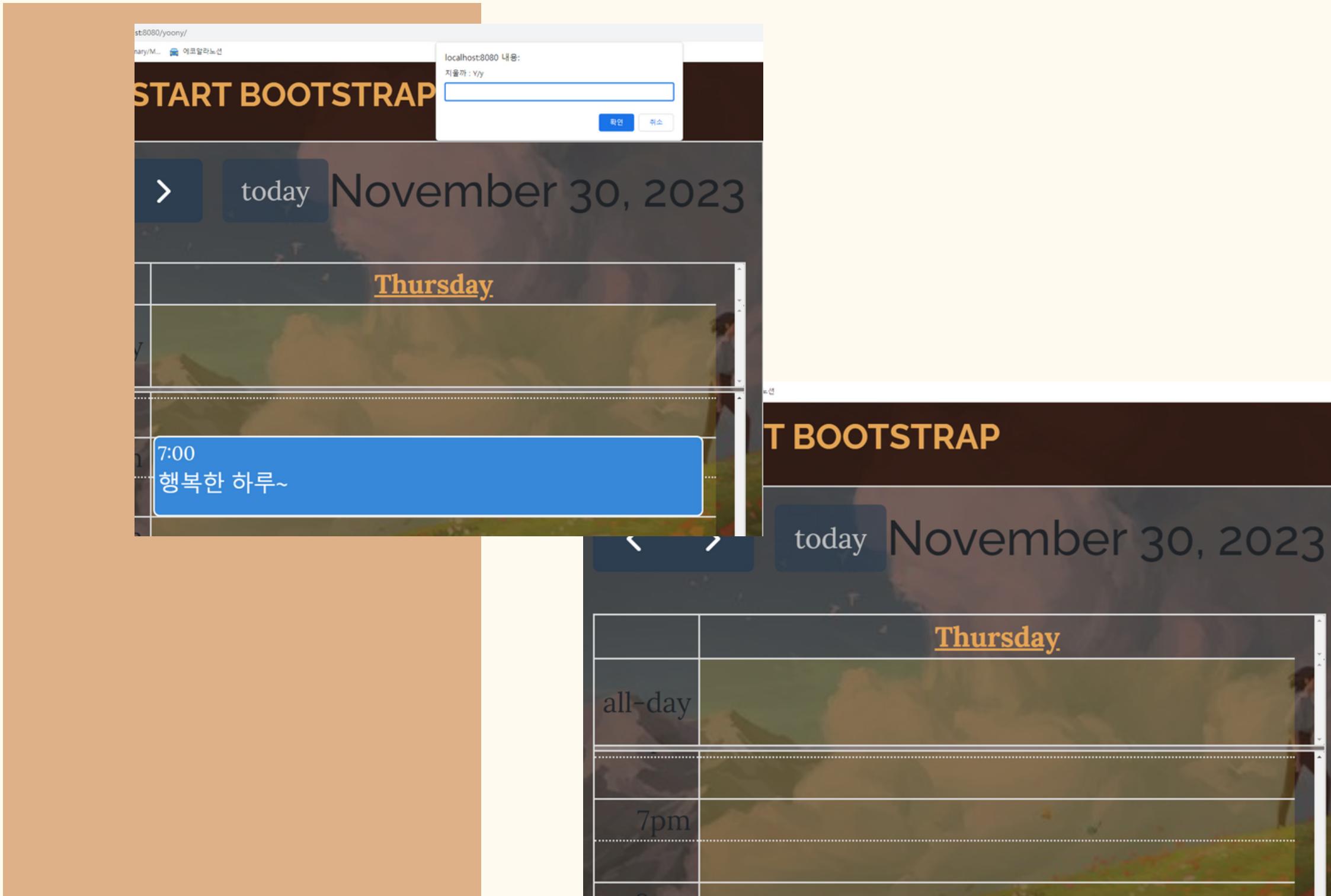


아이콘 사용

회원은 아이콘을 클릭하면
해당 시간의 시간대에
아이콘에 맵핑한 코멘트가 저장

캘린더에 출력

02 주요 기능

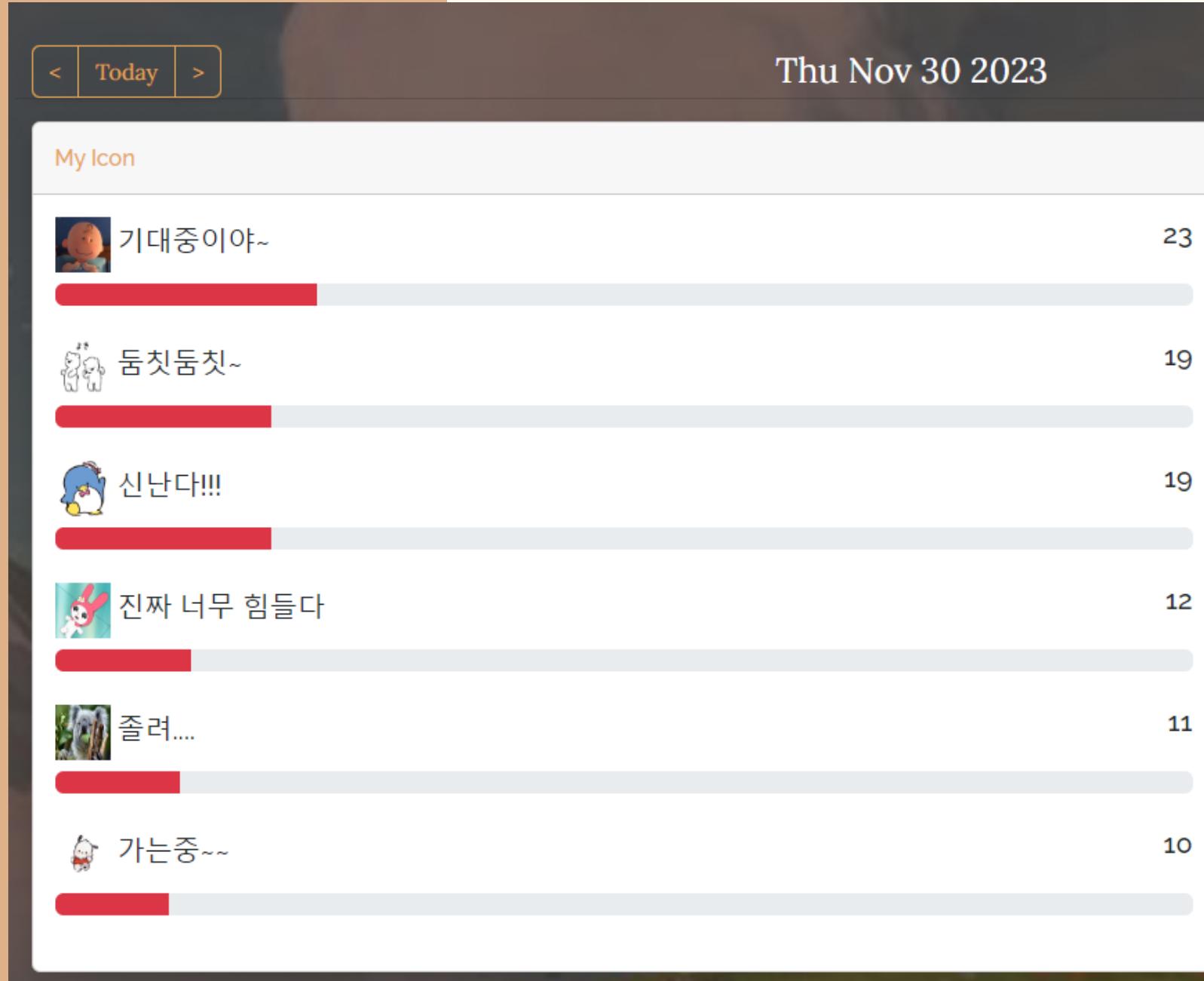


아이콘 사용

회원은 아이콘을 클릭하면
해당 시간의 시간대에
아이콘에 맵핑한 코멘트가 저장

캘린더에 출력

02 주요 기능



마이페이지

회원의 아이콘 사용
횟수 시각화

03 프로젝트 핵심 소스 / 메인페이지



```
@RequestMapping(value = "/", method = RequestMethod.GET)
public String home(Locale locale, Model model, HttpSession session) throws Exception {
    logger.info("Welcome home! The client locale is {}.", locale);

    Date date = new Date();
    DateFormat dateFormat = DateFormat.getDateInstance(DateFormat.LONG, DateFormat.LONG, locale);

    String formattedDate = dateFormat.format(date);
    model.addAttribute("serverTime", formattedDate);

    if (session.getAttribute("login") != null) {
        MemberVO login = (MemberVO) session.getAttribute("login");
        IconVO iconVO = new IconVO();
        iconVO.setMemId(login.getMemId());
        List<IconVO> homeIconList = iconService.getHomeIcon(iconVO);
        model.addAttribute("getHomeIcon", homeIconList);
    }

    return "home";
}
```

메인 화면의 아이콘/캘린더 기능
로그인한 사용자만 이용 가능

03 프로젝트 핵심 소스 / 회원가입



```
<bean id="passwordEncoder"
class="org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder"/>
```

회원가입 컨트롤러

```
@RequestMapping("/registDo")
public String registDo(HttpServletRequest request) {
    String id = request.getParameter("id");
    String pw = passwordEncoder.encode(request.getParameter("pw"));
    String nm = request.getParameter("nm");
    System.out.println(pw);
    MemberVO member = new MemberVO(id, pw, nm);
    try {
        memberService.registMember(member);
    } catch (Exception e) {
        e.printStackTrace();
        return "errorView";
    }
    return "redirect:/";
}
```

회원가입 컨트롤러

사용자의 비밀번호는 암호화되어
데이터베이스 저장

03 프로젝트 핵심 소스 / 로그인



```
memberVO가 맵핑되어 돌아오면 리다이렉트
@RequestMapping("/loginDo")
public String loginDo(MemberVO member, HttpSession session
    , boolean remember, String fromUrl, HttpServletResponse response) {
    System.out.println(member);
    MemberVO login = memberService.loginMember(member);
    boolean match = passwordEncoder.matches(member.getMemPw(), login.getMemPw());
    if(login == null || !match) {
        로그인정보가 없으므로 나중에 처리하기 위해 msg=N
        return "redirect:/loginView?msg=N";
    }
    session.setAttribute("login", login);
    세션을 담고 아래에서 체크
    if(remember) {
        //true 쿠키 생성
        Cookie cookie = new Cookie("rememberId", member.getMemId());
        생성한 쿠키를 response(응답하는 객체)에 담아서 전달.
        response.addCookie(cookie);
    }else {
        // 쿠키 삭제
        Cookie cookie = new Cookie("rememberId", "");
        cookie.setMaxAge(0);
        response.addCookie(cookie); // 응답하는 객체에 담아서 전달
    }
    return "redirect:" + fromUrl;
}
```

로그인 컨트롤러

사용자의 id/pw 입력과 DB가
대응 여부에 따라

로그인 -> 이전 url로
|| 실패

03 프로젝트 핵심 소스 / 로그아웃



로그아웃 컨트롤러 세션종료시키는 역할

```
@RequestMapping("/logoutDo")
public String logoutDo(HttpServletRequest session, HttpServletRequest request) {
    // 세션종료
    session.invalidate();
    // 현재 요청이 어느 URL을 바라보는지
    String requestToUrl = request.getRequestURI().toString();
    String returnUrl = request.getHeader("Referer");
    System.out.println(returnUrl);
    return "redirect:" + returnUrl;
}
```

로그아웃 시
세션 만료

03 프로젝트 핵심 소스 / 마이페이지



```
@RequestMapping("/mypage")
public String mypage(HttpServletRequest session, Model model) throws Exception {
    //로그인한 사람만 들어올수있도록 세션 가져오기
    if (session.getAttribute("login") == null) {
        return "redirect:/loginView";
    }
    MemberVO login = (MemberVO) session.getAttribute("login");
    model.addAttribute("member", login);
    // 마이페이지 입장시 countCldPerIcon 데이터 가져오기
    List<Map<String, Object>> countCldPerIcon = cldService.countCldPerIcon(login);
    // 화면에서 foreach로 접근 가능
    model.addAttribute("countCldPerIcon", countCldPerIcon);

    return "member/mypage";

public List<Map<String, Object>> countCldPerIcon(MemberVO vo) throws Exception{
    List<Map<String, Object>> result = dao.countCldPerIcon(vo);
    if (result == null) {
        throw new Exception();
    }
    return result;
}
```

마이 페이지 입장 시

아이콘 별 기록 총 횟수

03 프로젝트 핵심 소스 / 로그인 성공 메인페이지



```
@RequestMapping("/getHomeIcon")
public String homeIconList(Model model, HttpSession session) throws Exception {
    if (session.getAttribute("login") != null) {
        MemberVO login = (MemberVO) session.getAttribute("login");
        IconVO iconVO = new IconVO();
        iconVO.setMemId(login.getMemId());
        List<IconVO> homeIconList = iconService.getHomeIcon(iconVO);
        model.addAttribute("getHomeIcon", homeIconList);
    }
    return "home";
}
```

로그인 한 회원은
기존에 생성한

아이콘 리스트와 캘린더 기록 출력

```
@ResponseBody
@RequestMapping("/getHomeCld")
public List<IconViewVO> getHomeCldList(HttpSession session) throws Exception {
    if (session.getAttribute("login") != null) {
        MemberVO login = (MemberVO) session.getAttribute("login");
        CldVO cldVO = new CldVO();
        cldVO.setMemId(login.getMemId());
        List<IconViewVO> homeCldList = cldService.getHomeCLD(cldVO);
        return homeCldList;
    }
    return Collections.emptyList(); // 로그인되지 않았을 경우 빈 리스트 반환
}
```

03 프로젝트 핵심 소스 / 아이콘 등록(1)



```
@PostMapping("/files/upload")
public ResponseEntity<?> uploadFiles(@RequestParam("uploadImage") MultipartFile uploadImage,
    HttpServletRequest req, IconVO icon,
    @RequestParam Map<String, Object> map,
    HttpSession session) throws Exception {
    System.out.println(icon);
    String webPath = "/resources/iconImage/";
    String folderPath = session.getServletContext().getRealPath(webPath);
    System.out.println(folderPath);
    IconVO result = iconService.uploadProfile(icon, folderPath, webPath, uploadImage);
    Map<String, Object> response = new HashMap<>();
    response.put("message", "success");
    response.put("imagePath", result.getIconImg());
    response.put("iconNo", result.getIconNo());
    response.put("comment", icon.getIconComment());
    return new ResponseEntity<?>(response, HttpStatus.OK);
}
```

```
<button id="save-button"
    onclick="fn_icon('${s
    style="flex: 1; color
    등록</button>
```

```
/* 로그인 후 이미지&아이콘을 등록, DB에 저장 */
function fn_icon(p_id) {
    if (p_id == '') {
        alert("아이콘 등록은 로그인 이후 작성 가능합니다");
        location.href = '<c:url value="/loginView" />';
        return;
    }
    const imageFile = imageUploadInput.files[0];
    const comment = commentInput.value;
    if (!(imageFile && comment)) {
        alert("이미지와 코멘트를 모두 입력하세요.");
        return;
    }
    let formData = new FormData($('#iconForm')[0]);
    /* console.log(formData); */
    $.ajax({
        url: '<c:url value="/files/upload" />'
        , type: 'POST'
        , data: formData
        , dataType: 'json'
        , processData: false
        , contentType: false
    });
}
```

아이콘 등록 버튼

화면에서 사용자에게 입력받은
아이콘 이미지와 코멘트를
ajax & HttpServletRequest
활용

Database 저장 및
클라이언트에게
업로드가 성공됐음을 전송

03 프로젝트 핵심 소스 / 아이콘 등록(2)



```
let formData = new FormData($('#iconForm')[0]);
$.ajax({
  url: '<c:url value="/files/upload" />',
  type: 'POST',
  data: formData,
  dataType: 'json',
  processData: false,
  contentType: false,
  success: function (res) {
    console.log(res);
    /* 새롭게 생성된 icon table로 수정한 부분 */
    const tableRow = document.createElement("tr");
    tableRow.id = res.iconNo;

    const tableDataImage = document.createElement("td");
    const img = document.createElement("img");
    img.src = URL.createObjectURL(imageFile);
    img.alt = comment;
    img.style.cursor = "pointer";
    img.style.width = "40px";
    img.style.height = "40px";
    img.style.marginLeft = "7px";
    img.classList.add("icon-image")
    tableDataImage.appendChild(img);
    tableRow.appendChild(tableDataImage);

    const tableDataComment = document.createElement("td");
    tableDataComment.style.color = "rgb(251, 237, 219)";
    tableDataComment.style.fontSize = "15px";
    tableDataComment.className = "icon-comment";
    console.log(tableDataComment);
    tableDataComment.textContent = comment;
    tableRow.appendChild(tableDataComment);

    const tableDataDelete = document.createElement("td");
    const deleteButton = document.createElement("a");
    deleteButton.textContent = "X";
    deleteButton.style.cursor = "pointer";
    deleteButton.onclick = function() {
      fn_del_icon(res.iconNo);
    };
    tableDataDelete.appendChild(deleteButton);
    tableRow.appendChild(tableDataDelete);

    iconBody.appendChild(tableRow);
    // 입력 필드 초기화
    imageUploadInput.value = "";
    commentInput.value = "";
  },
  error: function (e) {
    console.log(e);
  }
});
```

아이콘 등록 버튼

성공 시

새로 생성한 아이콘
아이콘 리스트에
요소 추가

[이미지, 코멘트, 삭제버튼]

03 프로젝트 핵심 소스 / 아이콘 리스트



```
<!-- 등록한 아이콘 출력부분 -->
<div class="all-saved-item">
    <!-- 새롭게 추가하는 아이콘 -->
    <!-- <div class="saved-items" id="savedItems"> </div> -->
    <table class="table">
        <tbody id="iconBody">
            <c:forEach items="${getHomeIcon }" var="icon">
                <tr id="${icon.iconNo }">
                    <td>
                        
                    </td>
                    <td style="color: rgb(251, 237, 219); font-size: 15px;" class="icon-comment">${icon.iconComment}</td>
                    <!-- 삭제버튼 -->
                    <td style="color: rgb(251, 237, 219); "><a onclick="fn_del_icon('${icon.iconNo} ')>x</a></td>
                </tr>
            </c:forEach>
        </tbody>
    </table>
</div>
```

등록한 아이콘 리스트

기존 DB에 저장된 아이콘 출력

하단에 새로 생성한 아이콘 출력

03 프로젝트 핵심 소스 / 아이콘 클릭 이벤트



```
var isoTime = currentTime.toISOString();

$(document).ready(function () {
    $("#iconBody tr").click(function () {
        alert($(this).find("img"));
        var comment = $(this).find("td:nth-child(2)").text();
        let make_img = $(this).find("img").clone();
        let formdata = {
            memId: $("#memId").val()
            , iconNo: $(this).attr("id")
            , iconComment: comment
        }
        console.log("memId: " + formdata.memId);
        console.log("iconNo: " + formdata.iconNo);
        console.log("iconComment: " + formdata.iconComment);
        $.ajax({
            url: '<c:url value="writeCldDo/" />'
            , type: 'POST'
            /* post 방식으로 보낼 컨텐츠 타입 */
            , contentType: 'application/json'
            , dataType: 'json'
            , data: JSON.stringify(formdata)
            , success: function (res) {
                console.log(res);
            }, error: function (e) {
                console.log(e);
            }
        });
        const checkEvent = {
            title: comment,
            start: isoTime,
        };
        calendar.addEvent(checkEvent);
    });
    savedItems.addEventListener("click", function (event) {
        const target = event.target;
        if (target.tagName === "IMG") {
            const checkEvent = {
                title: target.parentElement.querySelector(".comment-item").textContent,
                start: isoTime
            };
            calendar.addEvent(checkEvent);
        }
    });
});
```

아이콘 클릭 시

DB cld테이블(달력 기록)에 추가
FullCalendar 화면에 이벤트 추가

03 프로젝트 핵심 소스 / FullCalendar



```
document.addEventListener('DOMContentLoaded', function () {
    // FullCalendar 코드
    var calendarEl = document.getElementById('calendar');
    calendar = new FullCalendar.Calendar(calendarEl, {
        initialView: 'timeGridDay',
        headerToolbar: {
            left: 'prev,next today',
            center: 'title',
            right: ''
        },
        events: function (fetchInfo, successCallback, failureCallback) {
            // 이벤트 데이터를 서버에서 비동기적으로 가져옴
            $.ajax({
                url: '<c:url value="/getHomeCLD" />', // 서버에서 일정 데이터를 가져올 엔드포인트
                type: 'GET',
                dataType: 'json', // 가져온 데이터 형식에 따라 조정
                success: function (data) {
                    // 서버에서 받은 데이터를 FullCalendar에 추가
                    var events = data.map(function (eventData) {
                        return {
                            title: eventData.iconComment, // 이벤트 제목
                            start: eventData.cldTime // 이벤트 시작 시간 (날짜 및 시간 형식)
                        };
                    });
                    successCallback(events);
                },
                error: function (error) {
                    console.error('일정 데이터를 가져오는 중 오류 발생:', error);
                    failureCallback(error);
                }
            });
        },
        eventClick: function (res) {
            let action = prompt("지울까 : Y/y");
            if (action == 'Y' || action == 'y') {
                res.event.remove();
            } else { }
        },
    });
    calendar.render();
});
```

FullCalendar

DB의 기존 데이터 연동과
이벤트 클릭 시 삭제 기능

03 프로젝트 핵심 소스 / 마이페이지



```
<div class="card-body">
    <c:forEach items="${countCldPerIcon }" var="icon">
        <h4 class="small font-weight-bold" style="font-size: 20px">
            
            ${icon.ICON_COMMENT}
            <span class="float-right" style="float: right;">${icon.COUNTCLDPERICON}</span>
        </h4>
        <div class="progress mb-4">
            <div class="progress-bar bg-danger"
                role="progressbar"
                style="width: ${icon.COUNTCLDPERICON}%">
                ${icon.COUNTCLDPERICON}
            <div style="width: 100%; position: relative; height: 0; border-bottom: 1px solid #ccc; margin-top: -10px;">
                <div style="position: absolute; left: 50%; bottom: -10px; width: 0; height: 0; border-left: 10px solid transparent; border-right: 10px solid transparent; border-top: 20px solid #ccc;"></div>
            </div>
        </div>
    </c:forEach>
</div>

<!-- 아이콘마다의 cld 기록 횟수 --&gt;
&lt;select id="countCldPerIcon" parameterType="MemberVO" resultType="java.util.Map" &gt;
    SELECT
        a.icon_no
        , a.icon_img
        , a.icon_comment
        , COUNT(b.cld_no) as countCldPerIcon
    FROM user_icon a, calendar b
    WHERE a.icon_no = b.icon_no
    AND a.del_yn = 'N'
    AND b.cld_del_yn = 'N'
    GROUP BY( a.icon_no
        , a.icon_img
        , a.icon_comment)
    ORDER BY countCldPerIcon DESC
&lt;/select&gt;</pre>
```

마이페이지

아이콘마다의 달력 기록 횟수를
아코디언 형식의
막대 그래프로 출력

```
@RequestMapping("/mypage")
public String mypage(HttpServletRequest session, Model model) throws Exception {
    if (session.getAttribute("login") == null) {
        return "redirect:/loginView";
    }
    MemberVO login = (MemberVO) session.getAttribute("login");
    model.addAttribute("member", login);
    List<Map<String, Object>> countCldPerIcon = cldService.countCldPerIcon(login);
    model.addAttribute("countCldPerIcon", countCldPerIcon);
    return "member/mypage";
}
```

04 향후 개선점



직관적인 UI

처음 진행하는 프로젝트여서
어떤 형태의 UI가 보다 컨텐츠를 이용하기
쉬울지 고려하지 못한 부분을 개선

다양한 통계

사용자의 달력 기록과 관련된
감정 패턴을 비율, 시간대 등의
다양한 그래프로 시각화 추가

아이콘의 다양화

이모지 유니코드를 활용해
사용자에게 아이콘의 선택지를
제공하는 방식으로 다양화

보다 편리한 접근

‘아이콘으로 간편하게’라는
서비스의 목적에 맞게
간략한 모바일 서비스 구현



**THANKS
FOR
WATCHING**

iove353395@gmail.com

<https://github.com/Yoon1717/YoonyCalendar.git>