



외부 API 활용 웹 개발 프로젝트

소원을 말하는 MES 시스템

목차

- 팀원 소개
- 동기
- 개발환경
- 스킬
- 스케줄
- UML
- 스토리보드

- ERD
- 구현 화면 Sample
- 코드 Sample
- 후기



팀원 소개

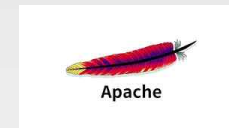
- ◉ 정근승(팀장) : 전체적인 통괄 및 정보/BOM 페이지 서블릿 구현
- ◉ 권대호 : 생산관리 및 메인페이지 서블릿 구현
- ◉ 정다올 : 설비관리 서블릿 구현
- ◉ 김소원 : 품질관리 서블릿 구현
- ◉ 김진홍 : 재고관리 및 로그인/회원가입 및 관리자 페이지, 게시글 페이지 서블릿 구현

동기

- ◉ 배관 제조 업체
- ◉ 1차 프로젝트에서 프론트엔드로 구현한 사이트를 보완하고 서블릿으로 교체하기 위해 그대로 진행하였습니다.

개발환경

- ◉ 운영체제(OS) : window 10 Home 64bit
- ◉ 통합 개발환경(IDE) : Eclipse(Java development Kit ver.11), VSCode (HTML, CSS, JavaScript)
- ◉ 데이터 베이스(DB) : Oracle SQL developer
- ◉ 웹 애플리케이션 서버(WAS) : Tomcat ver.9
- ◉ 형상 관리 (Version Control) : SourceTree, GitHub
- ◉ 기타 : Miro(스토리보드와 UML), 구글 스프레드시트(간트차트)



스킬 습득

- Java/JSP/Servlet : 로직을 처리하는 백엔드 코드를 작성하고 클라이언트와 서버간의 통신을 구현하는 스킬
- API 개발 : 클라이언트와 서버 간 데이터를 주고 받기 위한 RESTful API를 설계하고 구현하는 스킬

- AJAX/Fetch : JavaScript를 이용하여 비동기적으로 서버와 데이터를 주고받는 기술
- JSON 파싱 : 서버로부터 받은 JSON 데이터를 처리하고, 프론트엔드에서 이를 활용하는 기술

- ERD 설계 : DB 테이블 구조와 관계를 정의하는 다이어그램 작성 향상
- DB 최적화 : 데이터베이스 성능을 높이기 위해 인덱싱 및 쿼리 최적화 스킬

- 팀워크 및 의사소통 : 팀원들과의 협업을 통해 효과적인 커뮤니케이션 방법과 문제 해결 능력
- 디버깅 : 개발 중 발생하는 오류를 분석하고, 디버깅을 통해 문제를 해결하는 스킬 향상

스케줄

- 1주차: 기획 및 초기 설계

- 프로젝트 요구 사항 확정
- 데이터베이스 설계 및 ERD 다이어그램 작성
- 기본 백엔드 구조 설계 (Controller, Service, DAO 구조)

- 3주차: 테스트 및 최종 조정

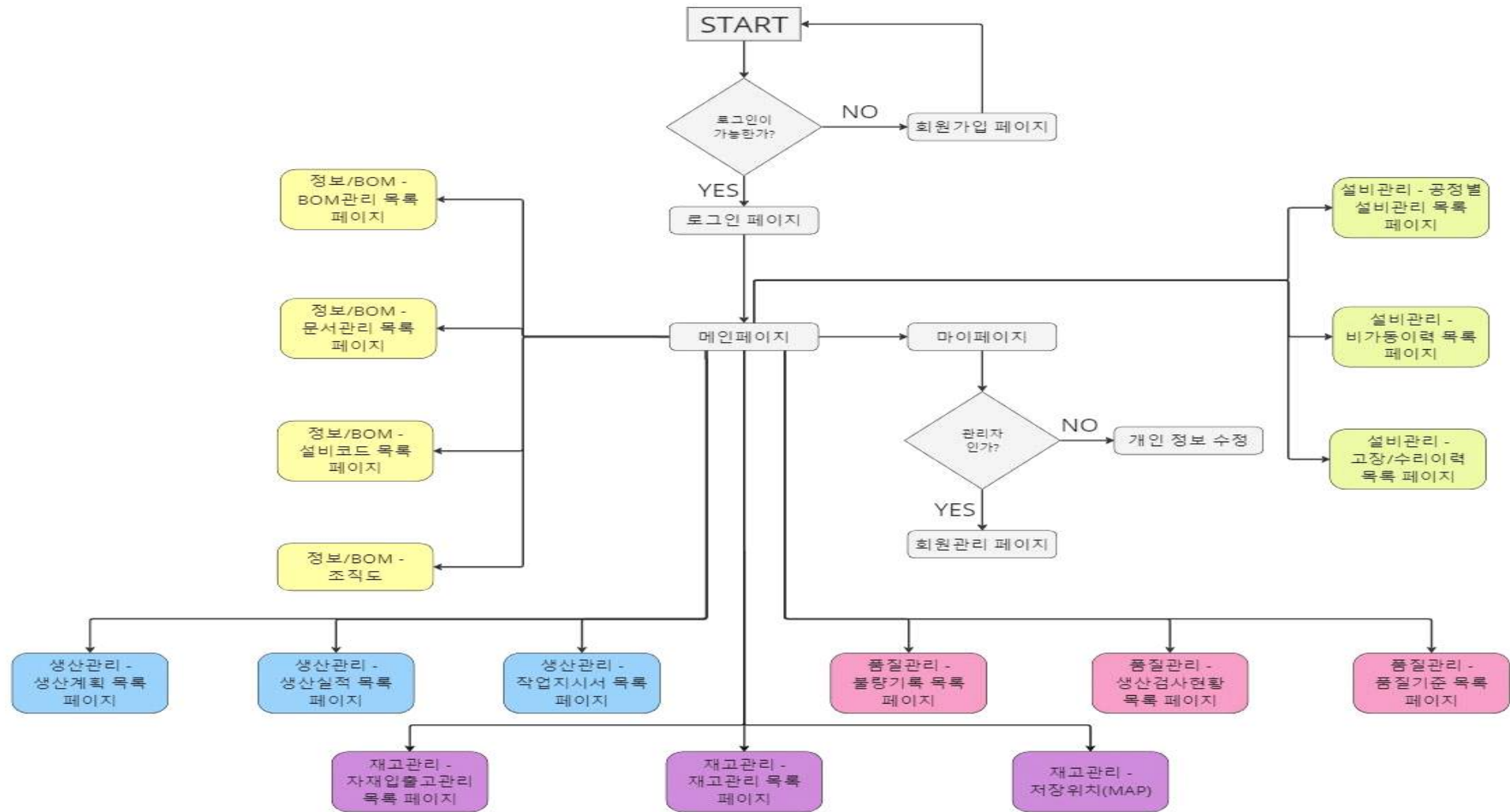
- 통합 테스트
- 버그 수정 및 최종 피드백 반영

- 2주차: 개발 및 연동

- 데이터베이스 구축 및 초기 데이터 삽입
- API 및 백엔드 로직 구현
- 프론트엔드-백엔드 연동

[illegible]

UML



스토리보드

네인/게시판

로그인 페이지



관리자페이지



재고관리 - 사...



생산관리 - 생...



정보/BOM - D...



품질관리 - 제품...



설비관리 - 설비...



메인페이지



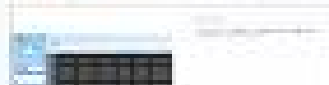
공지사항



재고관리 - 재...



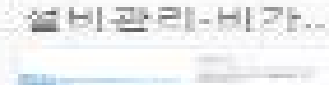
생산관리 - 생...



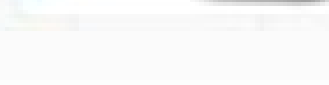
정보/BOM - 생...



품질관리 - 생...



설비관리 - 비가...



마이페이지



외원가입(popup)



재고관리 - 지...



생산관리 - 생...



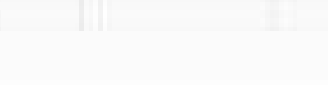
정보/BOM - 조...



품질관리 - 감...



설비관리 - 감...



게시글 작성(p...



모달창(모든해...



생산관리 - 생...



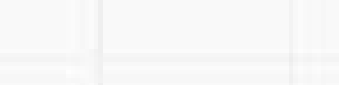
정보/BOM - 생...



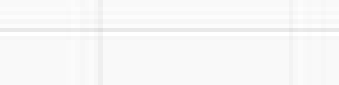
품질관리 - 생...



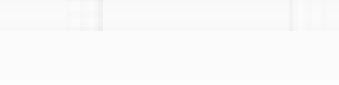
설비관리 - 생...



정보/BOM - 생...



품질관리 - 생...



재고관리

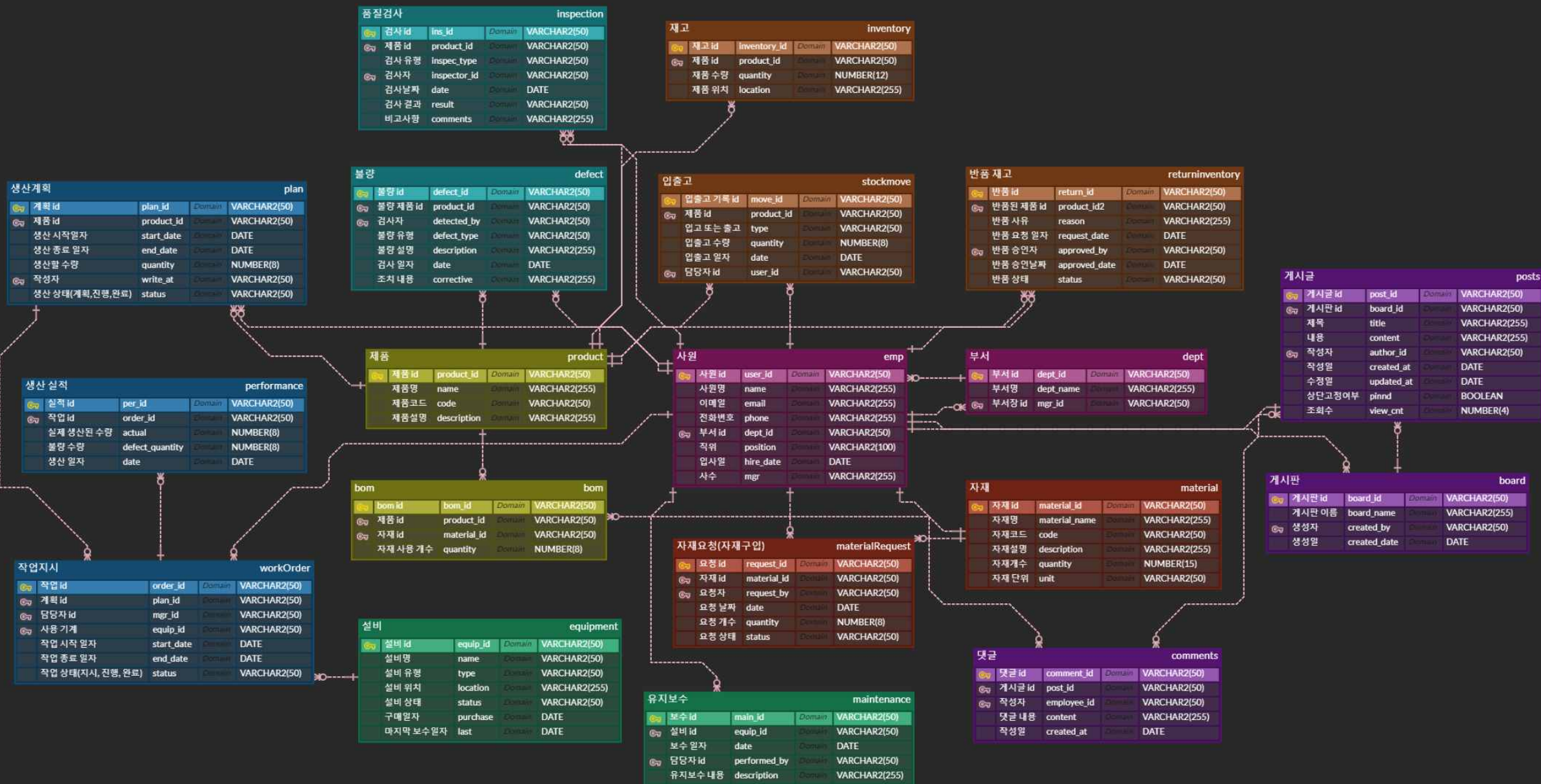
생산관리

정보/BOM

품질관리

설비관리

ERD



A decorative graphic consisting of several overlapping circles of different sizes and colors (green, blue, teal) on a dark blue background. The circles are filled with a fine grid pattern. The background is split horizontally into a dark blue top half and a light blue bottom half.

[illegible]

정근승 코드 Sample

선택된 체크박스의 tr을 구하고 그 안 기본키로
할당된 코드에 아이디를 줘서 그 아이디를
배열에 담아 Ajax에 보낼때 심표로 구분하여
보냅니다. 보낸 코드를

```
function delchk() {
    if (!confirm("정말로 삭제하시겠습니까?")) {
        event.preventDefault();
    } else {
        let selectchk = document.querySelectorAll('.selectchk');
        let bomIDs = [];

        // 체크된 체크박스에 대한 docID 수집
        for (let checkbox of selectchk) {
            if (checkbox.checked) {
                let row = checkbox.closest('tr'); // 체크박스가 포함된 <tr> 찾기
                let bomID = row.querySelector('#bomID'); // 해당 <tr> 내의 #docID
                if (bomID) {
                    bomIDs.push(bomID.textContent); // docID 값을 배열에 추가
                }
            }
        }

        // docID 값을 쉼표로 구분된 문자열로 서블릿에 전송
        if (bomIDs.length > 0) {
            let xhr = new XMLHttpRequest();
            xhr.open("POST", "/mes/BOM/deleteSelect", true); // 서블릿 URL로 POST
            xhr.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");

            // 데이터를 쉼표로 구분된 문자열 형식으로 전송
            xhr.send("bomIDs=" + encodeURIComponent(bomIDs.join(',')));

            xhr.onreadystatechange = function () {
                if (xhr.readyState === 4 && xhr.status === 200) {
                    // 서버 응답을 처리하는 부분 (성공시)
                    console.log("서버로부터 응답:", xhr.responseText);
                }
            };
        } else {
            alert("선택된 항목이 없습니다.");
        }

        window.location.href = "/mes/BOM/list";
    }
}
```

```
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    System.out.println("Doc Delete Select Controller doPost 실행");

    // 쉼표로 구분된 docIDs 파라미터 수신
    String bomIDsParam = request.getParameter("bomIDs");

    if (bomIDsParam != null && !bomIDsParam.isEmpty()) {
        // 쉼표로 구분된 문자열을 배열로 변환
        String[] bomIDArray = bomIDsParam.split(",");

        // 문자열 배열을 Integer 리스트로 변환
        List<String> bomIDList = new ArrayList<>();
        for (String id : bomIDArray) {
            bomIDList.add(id.trim()); // 문자열을 정수로 변환 후 리스트에 추가
        }

        System.out.println("받은 docID 리스트: " + bomIDList);

        // 서비스에 정수 리스트 전달
        BOM_Service service = new BOM_Service();
        int dto = service.deleteSelect(bomIDList);

        response.sendRedirect("list");
    }
```

다시 구분된 쉼표를 제거하여 배열로 변환한 뒤 오라클
SQL문을 반복 실행하여 원하는 만큼의 데이터가 지워지
게 했습니다.

김진홍 구현화면 Sample

홍길동

25

1234567890

September 2024

1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

자재관리

재고관리

주문관리

생산관리

입고관리

자재현황관리

자재의 입고를 기록해 판매 조달하고 중량관리 관리하는 페이지

필터 설정

100

추가

재고 정보 삭제

Start Date

2024-09-01

~

End Date

2024-09-30

검색

필터 적용

▶	자재번호	Lot번호	자재이름	수량(KG)	규격(mm)	위치	최신회달일	수정
▶	P001	LO1001	Product A	100	Spec A	Location 1	2024-09-01	수정
▶	P002	LO1002	Product B	200	Spec B	Location 2	2024-09-10	수정

+

김진홍 코드 Sample

```
filterByDateAndStatusButton.addEventListener("click", function () {
    let startDateInput = document.getElementById("startDate").value;
    let endDateInput = document.getElementById("endDate").value;

    // 모든 행을 숨긴 후 필터링된 행만 보이도록 설정
    for (let i = 0; i < rows.length; i++) {
        rows[i].style.display = 'none';
    }

    filteredRows = Array.from(rows).filter(row => {
        let dateCell = row.getElementsByTagName("td")[7].innerText;
        let rowDate = new Date(dateCell);

        // 필터링 조건: 날짜 범위 내에 있어야 함
        let dateMatches = (!startDateInput || !endDateInput) ||
            (rowDate >= new Date(startDateInput) &&
            rowDate <= new Date(endDateInput));

        return dateMatches;
    });

    setupPagination();
});
```

날짜에 필터를 걸어 원하는 날짜의 정보를 얻게 할 수 있는 기능입니다. 함수를 만들어 css 스타일에 display:none을 주워 필터링 하였습니다.

김소원 구현화면 Sample





홍길동
 관리자
 관리자용

주역자관리
 관리자관리

September 2024

1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

품질관리

품질관리 시험기록을 조회하는 페이지 입니다.

사용자 관리

사용자 관리

시험기록을 검색

검색

시험기록 ID

시험기록명

과목명


시험일자

시험시간

시험결과

시험기록 ID	시험기록명	과목명	시험일자	시험시간	시험결과
000001	품질관리 시험 1	과목명	2024-09-01	09:00-10:00	합격
000002	품질관리 시험 2	과목명	2024-09-02	09:00-10:00	합격
000003	품질관리 시험 3	과목명	2024-09-03	09:00-10:00	합격
000004	품질관리 시험 4	과목명	2024-09-04	09:00-10:00	합격
000005	품질관리 시험 5	과목명	2024-09-05	09:00-10:00	합격
000006	품질관리 시험 6	과목명	2024-09-06	09:00-10:00	합격
000007	품질관리 시험 7	과목명	2024-09-07	09:00-10:00	합격
000008	품질관리 시험 8	과목명	2024-09-08	09:00-10:00	합격
000009	품질관리 시험 9	과목명	2024-09-09	09:00-10:00	합격
000010	품질관리 시험 10	과목명	2024-09-10	09:00-10:00	합격

이전 | 다음



홍길동
 관리자
 관리자용

주역자관리
 관리자관리

September 2024

1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

생산검사현황

생산검사 현황을 조회하는 페이지입니다.

사용자 관리

사용자 관리

시험기록을 검색

검색

시험기록 ID

시험기록명

과목명


시험일자

시험시간

시험결과

시험기록 ID	시험기록명	과목명	시험일자	시험시간	시험결과
000001	생산검사 시험 1	과목명	2024-09-01	09:00-10:00	합격
000002	생산검사 시험 2	과목명	2024-09-02	09:00-10:00	합격
000003	생산검사 시험 3	과목명	2024-09-03	09:00-10:00	합격

이전 | 다음



홍길동
 관리자
 관리자용

주역자관리
 관리자관리

September 2024

1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

불량 기록

불량 기록 현황을 조회하는 페이지입니다.

사용자 관리

사용자 관리

시험기록을 검색

검색

시험기록 ID

시험기록명

과목명

시험일자

시험시간

시험결과

시험기록 ID	시험기록명	과목명	시험일자	시험시간	시험결과
000001	불량 기록 시험 1	과목명	2024-09-01	09:00-10:00	합격
000002	불량 기록 시험 2	과목명	2024-09-02	09:00-10:00	합격

이전 | 다음

김소원 코드 Sample

```
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException {
    request.setCharacterEncoding("utf-8");
    response.setContentType("text/html; charset=utf-8;");

    String quality_id = request.getParameter("quality_id");
    String title = request.getParameter("title");
    String mgr = request.getParameter("mgr");
    String insti = request.getParameter("insti");
    String revision = request.getParameter("revision");
    System.out.println(quality_id + ", " + title + ", " + mgr + ", " + insti + ", " + revision);

    LocalDate parsedRevision = null;
    if (revision != null && !revision.isEmpty()) {
        parsedRevision = LocalDate.parse(revision);
    }
    StandardDTO dto = new StandardDTO();
    dto.setQuality_id(quality_id);
    dto.setTitle(title);
    dto.setMgr(mgr);
    dto.setInsti(insti);
    dto.setRevision(parsedRevision);

    request.setAttribute("dto", dto);
    request.getRequestDispatcher("/WEB-INF/quality/standard/품질기준_Standard_modify.jsp")
}
```

각종 페이지별로
Controller, Service,
DTO, DAO 파일을 세분
화 하여 CRUD를 진행했
습니다.

권대호 코드 Sample

```
<script>
// "새 계획 추가" 버튼 클릭 시 모달 열기
document.getElementById('addPlanBtn').addEventListener('click', function() {
    document.getElementById('planForm').reset(); // 폼 초기화
    document.getElementById('actionType').value = 'add'; // action을 추가로 설정
    $('#planModal').modal('show'); // 모달 열기
});

// "수정" 버튼 클릭 시 모달 열기
document.querySelectorAll('.editPlanBtn').forEach(button => {
    button.addEventListener('click', function() {
        const planId = this.dataset.planid;
        // Ajax를 사용해 서버로부터 해당 plan 데이터를 받아옴
        $.ajax({
            url: '/production?action=getPlanById', // 수정할 생산 계획 정보를 가져오는 API
            method: 'GET',
            data: { planid: planId },
            success: function(plan) {
                // 받아온 데이터를 폼에 채워넣음
                document.getElementById('planid').value = plan.planid;
                document.getElementById('production_id').value = plan.production_id;
                document.getElementById('plannedQuan').value = plan.plannedQuan;
                document.getElementById('startDate').value = plan.startDate;
                document.getElementById('endDate').value = plan.endDate;
                document.getElementById('status').value = plan.status;
                document.getElementById('userid').value = plan.userid;

                document.getElementById('actionType').value = 'edit'; // action을 수정으로 설정
                $('#planModal').modal('show'); // 모달 열기
            }
        });
    });
});

// "저장" 버튼 클릭 시 폼 제출
document.getElementById('savePlanBtn').addEventListener('click', function() {
    document.getElementById('planForm').submit(); // 폼을 제출하여 추가 또는 수정 동작 수행
});
```

데이터 베이스로 읽어온 데이터들을 새로운 모달창에 값을 넣어 수정할수 있게 하였습니다.

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException {
    String action = request.getParameter("action");

    if ("add".equals(action)) {
        // 새로운 계획 추가 로직
        String production_id = request.getParameter("production_id");
        int plannedQuan = Integer.parseInt(request.getParameter("plannedQuan"));
        Date startDate = Date.valueOf(request.getParameter("startDate"));
        Date endDate = Date.valueOf(request.getParameter("endDate"));
        String status = request.getParameter("status");
        String userid = request.getParameter("userid");

        ProductionPlanDTO newPlan = new ProductionPlanDTO();
        service.addPlan(newPlan);
    } else if ("edit".equals(action)) {
        // 기존 계획 수정 로직
        String planid = request.getParameter("planid");
        String production_id = request.getParameter("production_id");
        int plannedQuan = Integer.parseInt(request.getParameter("plannedQuan"));
        Date startDate = Date.valueOf(request.getParameter("startDate"));
        Date endDate = Date.valueOf(request.getParameter("endDate"));
        String status = request.getParameter("status");
        String userid = request.getParameter("userid");

        ProductionPlanDTO updatedPlan = new ProductionPlanDTO();
        service.updatePlan(updatedPlan);
    }

    // 처리 완료 후 목록 페이지로 리다이렉트
    response.sendRedirect("/production");
}

private void displayPlanList(HttpServletRequest request, HttpServletResponse response) throws ServletException {
    // 목록 조회 로직
}
```

모달창에서 submit 하면 post 방식으로 넘어와 controller에 들어와서 getParameter 를 이용하여 수정되게 하는 Update 입니다.

정다올 구현 화면 Sample



홍길동
관리자

로그아웃

마이페이지

관리자페이지

September 2024						
일	월	화	수	목	금	토
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

정보/BOM 재고관리 품질관리 생산관리 설비관리

설비고장/수리이력

설비고장 또는 수리의 이력을 볼 수 있는 페이지입니다.

항목	내용
설비번호:	<input type="text"/>
설비명:	<input type="text"/>
설비유형:	<input type="text" value="검사"/>
설비위치:	<input type="text" value="위치"/>
구매일자:	<input type="text" value="연도-월-일"/>
상태:	<input type="text" value="상태"/>
고장/수리내용:	<input type="text"/>
관리자:	<input type="text"/>
고장/수리일자:	<input type="text" value="연도-월-일"/>
추가	<input type="button" value="추가"/>

게시물

별 조회 ~

<input type="checkbox"/>	번호	설비번호	설비명	설비유형	설비위치	구매일자	상태	고장/수리내용	관리자	고장/수리일자	비고
<input type="checkbox"/>	1	091	절단기	절단	공장 2	2024-08-25	운영중	점검	박민수	2024-09-01	
<input type="checkbox"/>	2	101	사출기	제조	공장 1	2019-08-22	점검중	금형교체	정다올	2024-08-31	
<input type="checkbox"/>	3	111	사출기	제조	위치	2023-01-02	상태	금형교체	정다올	2024-09-06	
<input type="checkbox"/>	4	123	사출기	제조	공장 1	2019-12-31	점검중	금형교체	정다올	2024-09-05	
<input type="checkbox"/>	5	equi002	레이저 절단기	절단	공장 2	2016-05-20	운영 중	레이저 절단기 정기 점검	김영희	2023-10-20	
<input type="checkbox"/>	6	equi003	조립 로봇	조립	공장 3	2017-07-25	점검 중	조립 로봇 수리	이철수	2023-11-25	
<input type="checkbox"/>	7	equi004	포장 기계	포장	공장 4	2018-09-30	운영 중	포장 기계 점검	박민수	2023-12-30	
<input type="checkbox"/>	8	equi004	포장 기계	포장	공장 4	2018-09-30	운영 중	포장 기계 점검	박민수	2023-04-20	
<input type="checkbox"/>	9	equi005	테스트 장비	검사	공장 5	2019-12-10	점검 중	테스트 장비 교체	정현수	2024-01-10	

정다올 코드 Sample

```
public void addEquipment(EquipmentDTO equipment) {
    String sql = "INSERT INTO equipment (equiID, equiname, equitype, selldate, equilloc, status) VALUES (?, ?, ?, ?, ?, ?)";
    String sqlMaintenance = "INSERT INTO maintenance (mainID, equiID, maindate, maincontent, manager) VALUES (?, ?, ?, ?, ?)";

    try (Connection cnt = getConnection();
        PreparedStatement ps = cnt.prepareStatement(sql);
        PreparedStatement psMaintenance = cnt.prepareStatement(sqlMaintenance)) {

        // equipment 테이블에 삽입
        ps.setString(1, equipment.getEquiID());
        ps.setString(2, equipment.getEquiname());
        ps.setString(3, equipment.getEquitype());
        ps.setDate(4, new java.sql.Date(equipment.getSelldate().getTime()));
        ps.setString(5, equipment.getEquilloc());
        ps.setString(6, equipment.getStatus());
        ps.executeUpdate();

        // maintenance 테이블에 삽입
        psMaintenance.setString(1, "main" + UUID.randomUUID().toString());
        psMaintenance.setString(2, equipment.getEquiID());
        psMaintenance.setDate(3, new java.sql.Date(equipment.getMaindate().getTime()));
        psMaintenance.setString(4, equipment.getMaincontent());
        psMaintenance.setString(5, equipment.getManager());
        psMaintenance.executeUpdate();

    } catch (SQLException e) {
        e.printStackTrace();
    }
}
```

```
protected void doPost(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    request.setCharacterEncoding("utf-8");
    response.setContentType("text/html; charset=utf-8");
    String equiID = request.getParameter("equiID");
    String equiname = request.getParameter("equiname");
    String equitype = request.getParameter("equitype");
    String equilloc = request.getParameter("equilloc");
    Date selldate = Date.valueOf(request.getParameter("selldate"));
    String status = request.getParameter("status");
    String maincontent = request.getParameter("maincontent");
    String manager = request.getParameter("manager");
    Date maindate = Date.valueOf(request.getParameter("maindate"));

    // 디버깅
    System.out.println("maindate: " + maindate);
    // 중복 확인
    EquipmentService equipmentService = new EquipmentService();
    if (equipmentService.getEquipmentById(equiID) != null) {
        request.setAttribute("errorMessage", "설비번호가 중복되었습니다.");
        request.getRequestDispatcher("/WEB-INF/equip/addEquipment.jsp").forward(request, response);
        return;
    }
    EquipmentDTO equipment = new EquipmentDTO();

    equipment.setEquiID(equiID);
    equipment.setEquiname(equiname);
    equipment.setEquitype(equitype);
    equipment.setEquilloc(equilloc);
    equipment.setSelldate(selldate);
    equipment.setStatus(status);
    equipment.setMaincontent(maincontent);
    equipment.setManager(manager);
    equipment.setMaindate(maindate);

    equipmentService.addEquipment(equipment);

    response.sendRedirect("Equip");
}
```

팝업창을 따로 띄워서 submit 하면 DAO 까지 들어오게 되고, 만들어야 될 데이터의 table이 두개이기 때문에 insert문을 2개를 작성하여, 따로 따로 prepare에 각각 세팅해주었습니다. 그렇게 데이터 베이스엔, 두개의 테이블에 새로운 데이터가 추가 되는 로직을 구현했습니다.

후기

○ 권대호

- 2차 프로젝트를 진행하면서 제 부족한 점을 많이 깨달을 수 있는 시간이었습니다. 1차 프로젝트 이후 저의 실력에 대해 느꼈지만, 이를 개선하기 위해 충분히 노력하지 않았습니다. 3차 프로젝트에서는 팀장과 팀원들의 의견에 귀 기울이며, 적극적으로 협력하고 팀의 목표를 위해 최선을 다하겠습니다. 이번 프로젝트를 통해 배운 점을 바탕으로 더 나은 팀원이 되도록 노력하겠습니다.

○ 정다올 후기

- 느낀점
- 1차때와 마찬가지로 개발자의 프로젝트란걸 제대로 느끼게 되었고 시간 분배의 중요성, 팀원들간의 소통이 중요하다는걸 알았습니다.
- 좋았던 점
- 흥내만 내던 페이지 기능들을 제대로 구현 해 볼수 있는 기회가 생겨 좋았습니다
- 어려웠던 점
- 서블릿에 대해 이해를 하지 못해 코딩에 진척이 없었으나 중반부부터 이해하여 초반에 하지 못한만큼 시간과 노력을 더 투자하여 따라갈 수 있게 하였습니다.
- 얻은 것
- 서블릿에 대한 이해도, 앞서 말씀 드렸던 시간 분배의 중요성 및 분배 방법 등을 습득 하게 되었습니다.

후기

◉ 김소원

- ◉ 모달을 처음 구현 해보는거여서 팀장님과 팀원들의 도움으로 이번에 알게 되어 성장을 한 기분이 들었습니다. 아직 제가 많이 부족하여 팀장님과 팀원들께 큰 도움은 못줘서 너무 죄송한 마음이 컸는데 다행이도 팀장님과 팀원들이 그런 저에게 먼저 다가와 더움을 주시고 할 수 있다는 용기를 주셔서 해 낼 수 있게 되었습니다.

◉ 김진홍

- ◉ 프로젝트에 뒤늦게 합류하면서 처음에는 적응하는 데 어려움이 있었지만, 팀원들의 도움 덕분에 점차 나아갈 수 있었습니다. 특히 프론트엔드와 백엔드를 연결하는 과정을 통해 웹 개발에 대한 이해도가 크게 향상되었습니다. 처음에는 각 부분이 어떻게 연결되는지 혼란스러웠지만, 팀원들과 협력하며 하나씩 해결해 나가면서 웹 애플리케이션의 구조와 흐름을 더 잘 이해하게 되었습니다. 덕분에 웹 개발에 대한 자신감도 많이 생기게 되었고, 팀의 일원으로서 기여할 수 있다는 점이 매우 뿌듯했습니다.

후기

◉ 정근승

- ◉ JSP와 Servlet을 사용해 HTML을 구현하는 과정이 쉽지 않았습니다. 처음에는 여러 파일을 세분화하지 않고 작업을 하다 보니 오류를 찾는 데 어려움을 겪었지만, DTO, DAO, Controller, Service 등으로 나누어 관리하면서 디버깅이 훨씬 수월해졌습니다. Controller에서 값을 설정해 JSP로 보내면 EL 태그로 데이터를 가져오는 과정에서 forEach를 사용해 값을 꺼내는 부분이 특히 헷갈렸습니다. 이를 해결하기 위해 선생님께 질문도 하고, 구글링을 통해 정보를 찾아보며 많은 복습을 했습니다. 이런 과정을 통해 조금씩 문제를 해결할 수 있었고, 큰 도움이 되었습니다.

Thank you for Watching!

