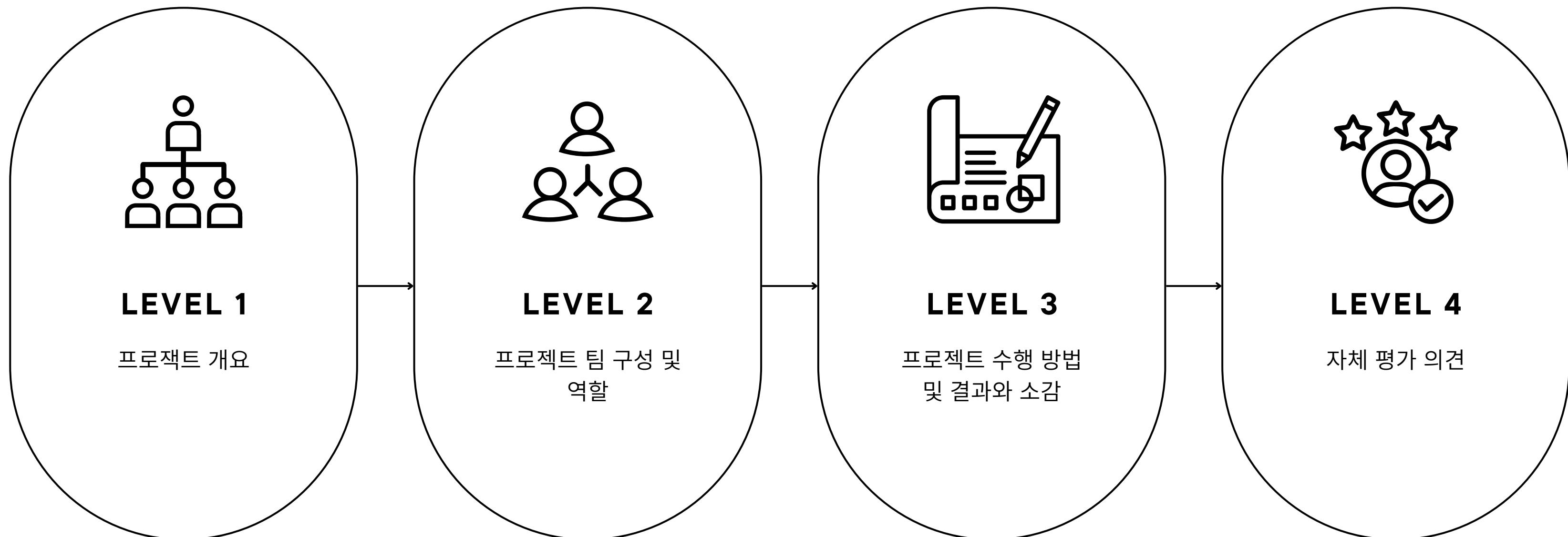


Cozy&Rest FURNITURE

배지현, 김다은, 이시연, 이현주, 정수하, 조다혜



목차

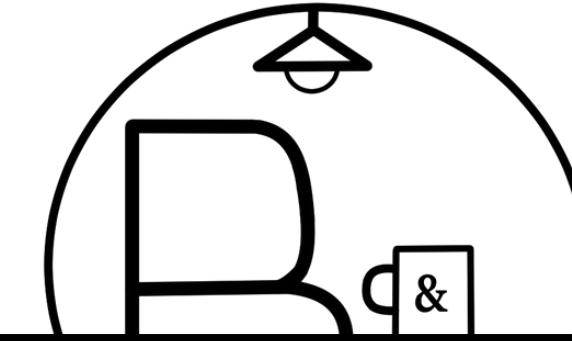


회사 소개



COZY&REST FURNITURE

아늑하고 편안한 공간을 만드는 가구



개발 주제선정 및 개요

가구를 생산하고 제조하는 회사의 MES 프로그램 개발

가구 제조업은 다른 산업에 비해 복잡한 공정과 다양한 작업 단계를 포함하며, 고가의 원재료와 제품을 사용하고 있어 생산 과정에서 오류나 불량은 큰 손실을 가져올 수 있습니다. 또한, 생산과 품질의 요구사항을 충족하고 자동화와 생산 효율이 필요하다고 생각해 가구 제조업을 위한 MES 프로그램을 개발했습니다.

개발 환경

OS

Window 10

WAS

Apache Tomcat (Version: 10.1.2)

WEB

java 17, Oracle DB
JSP, HTML5, CSS, JavaScript , Ajax, openAPI

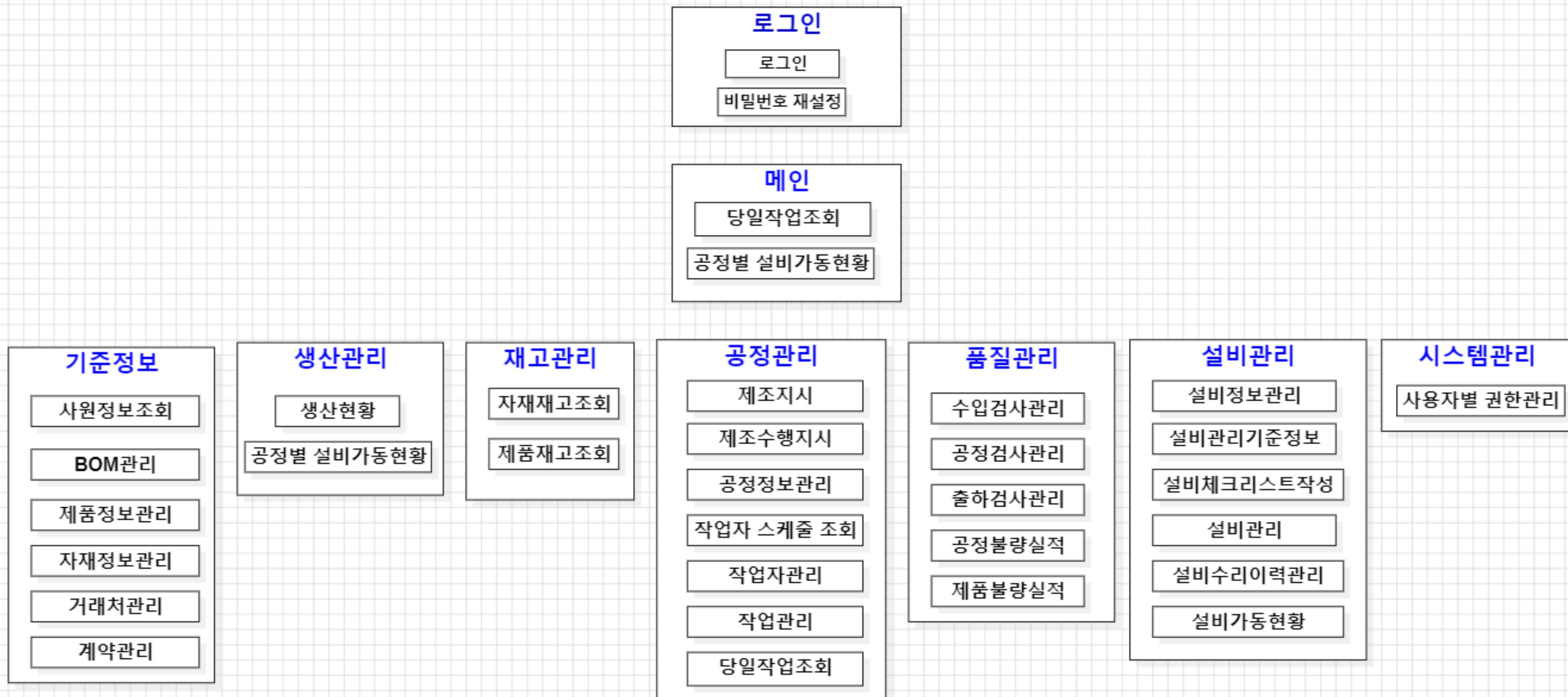
TOOL

IntelliJ, STS4(Eclipse), Git, GitHub, Figma, DBeaver, SQLDeveloper

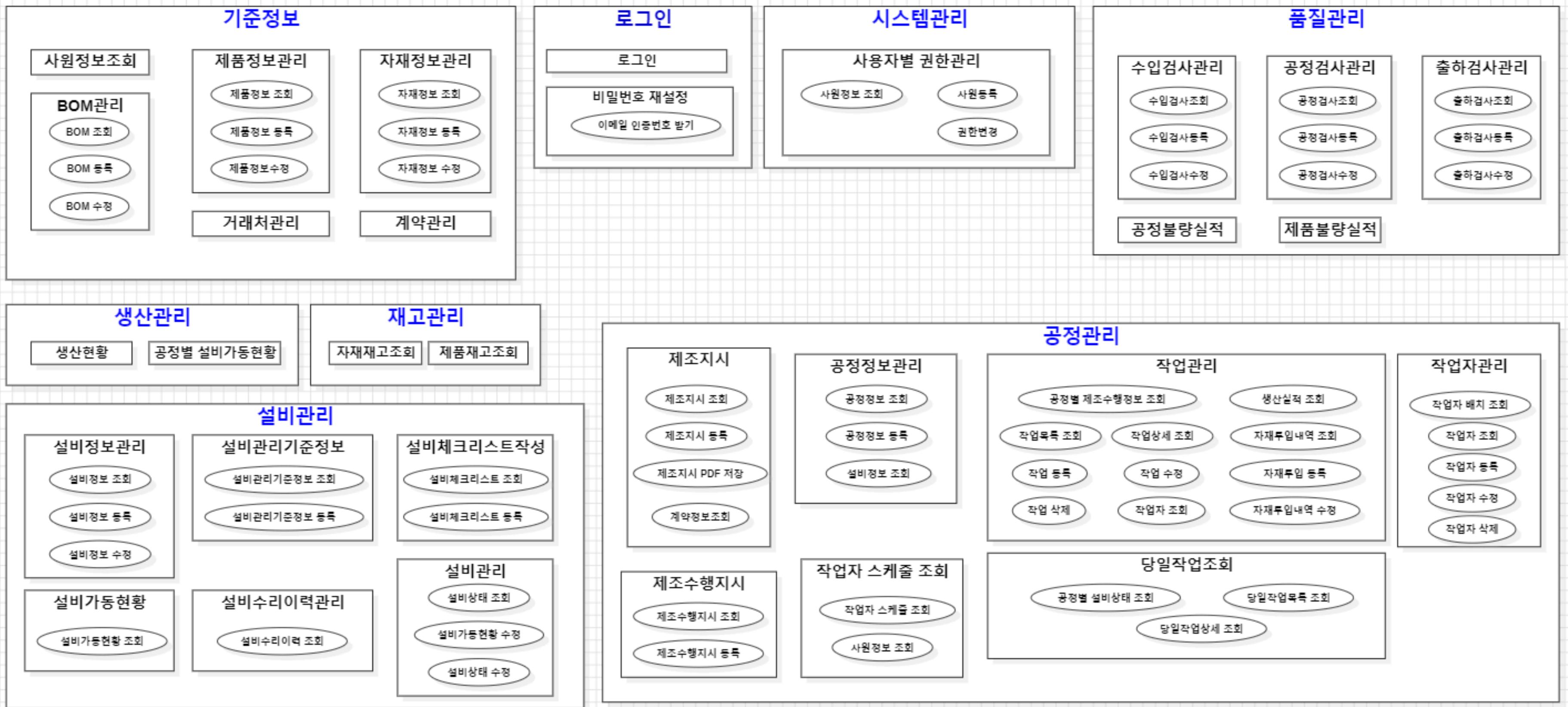
Open Source

fullcalendar(5.8), jsPDF (2.0.0), jsPDF AutoTable plugin (3.5.13), jQuery (3.7.1),
javax.mail (1.4.7), Mybatis (3.0.3), JSTL, Hikari(2.7.4)

화면구성도

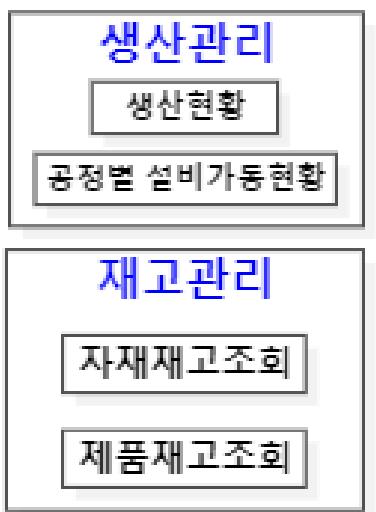
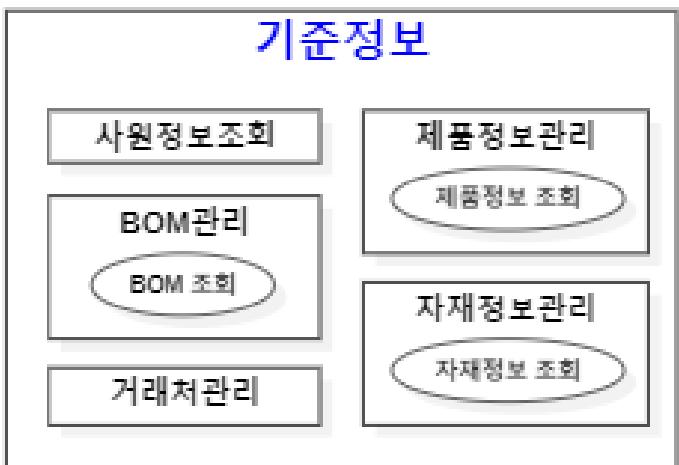


전체 유즈케이스 구성



권한별 유즈케이스 구성

사원



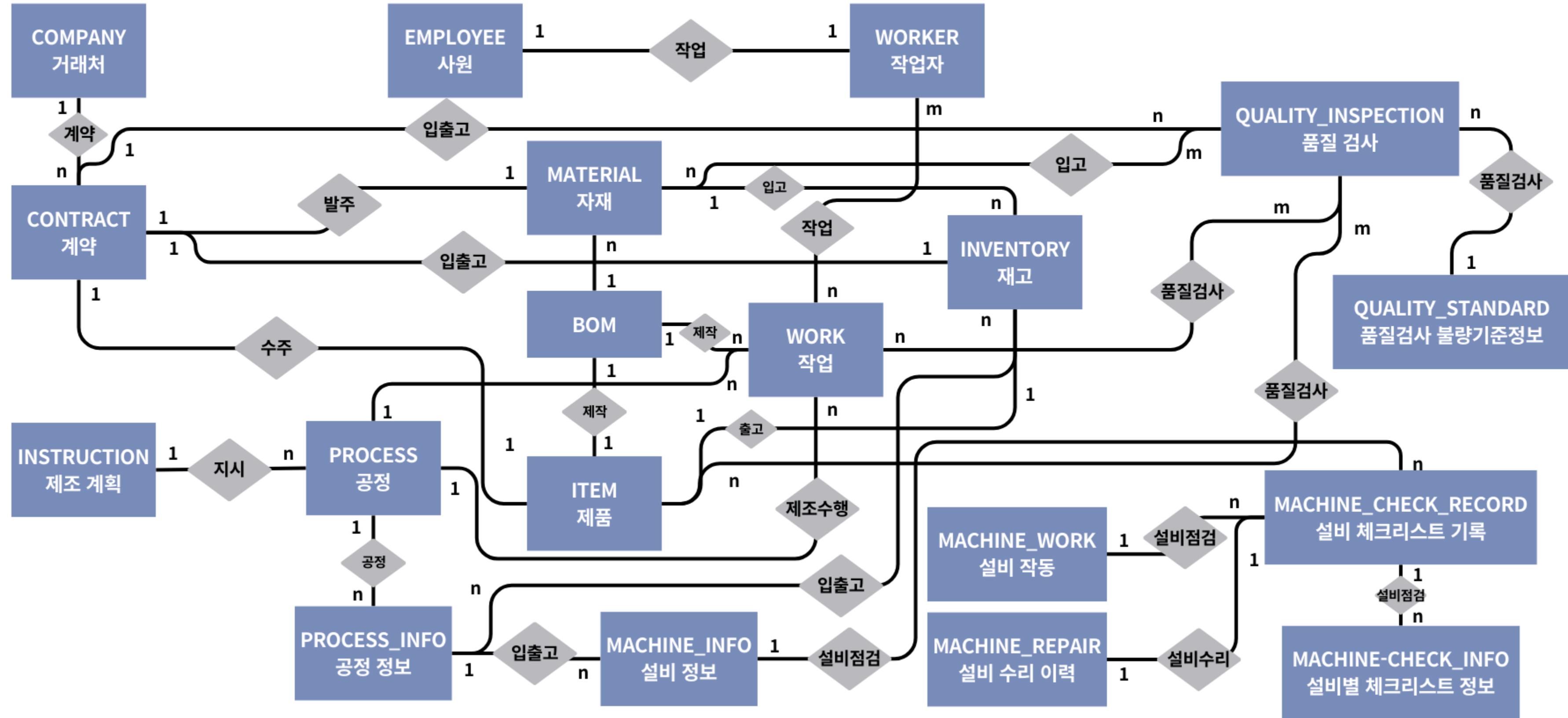
매니저



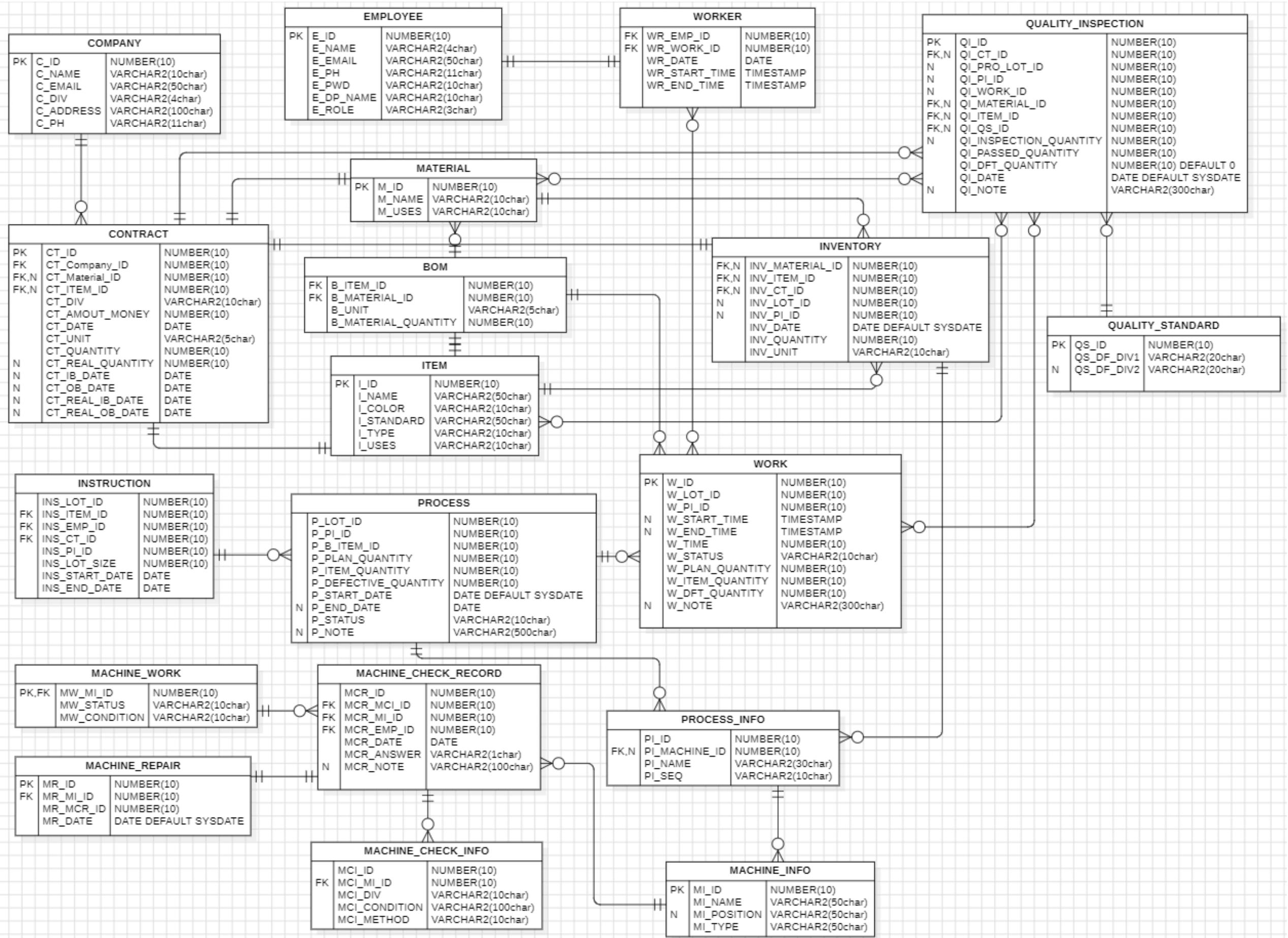
시스템관리

사원등록

개념적 데이터 모델



ERD - 논리적 데이터 모델링



기준정보

- 사원, 제품, 자재, BOM, 거래처, 계약

생산관리

- 제조계획, 제조수행, 공정정보, 작업, 설비정보, 설비작동

재고관리

- 제품, 자재, 재고

공정관리

- 계약, 제조계획, 제조수행, 공정정보, 작업, 사원, 회사, 작업자, BOM, 제품, 자재, 재고, 설비정보, 설비작동

설비관리

- 설비정보, 설비작동, 설비 체크리스트기록, 설비수리이력, 설비별 체크리스트 정보, 작업, 공정 정보

품질관리

- 품질검사, 품질검사 불량기준정보, 제품, 자재, 재고, 작업, 공정정보, 설비정보, 제조계획, 계약, 거래처

시스템관리

- 사원

C&R FURNITURE 개발 일정

프로젝트 팀 구성 및 역할

구성원	역할	담당업무
배지현	팀장 생산관리, 공정관리	<ul style="list-style-type: none">회의 진행 및 일정·팀원 관리, 전체 설계 담당RDBMS관리(OracleDB), 테이블 설계, 샘플데이터작성, ERD작성, 유즈케이스 작성공정관리 - 자재투입CRU, 작업CRUD, 작업자 CRUD생산관리 - 생산현황, 공정별설비가동현황
김다은	설비관리, 설비점검	<ul style="list-style-type: none">테이블 설계, 샘플데이터작성설비관리 - 설비정보 CRU, 설비관리 CRU, 설비수리이력조회, 설비가동현황조회설비점검 - 설비체크기준 CRU, 설비 체크리스트 CR
이시연	로그인 권한관리	<ul style="list-style-type: none">로고제작, 테이블 설계, 샘플데이터작성로그인(Spring Security), 인증번호를 이용한 비밀번호 재설정(이메일 인증번호 받기)권한관리 - 각 페이지 권한 부여, 사용정보 CRU
이현주	제품/자재 정보관리, BOM 관리	<ul style="list-style-type: none">테이블 설계, 샘플데이터작성기준정보 - 자재/제품정보 CRU, BOM CRU공정관리 - 작업자스케줄조회(풀캘린더 라이브러리), 구글캘린더
조다혜	공정관리	<ul style="list-style-type: none">테이블 설계, 샘플데이터작성공정관리 - 제조계획지시 CR & PDF 저장, 제조수행지시 CR, 공정정보 CR
정수하	자재/제품 재고 조회, 품질관리	<ul style="list-style-type: none">테이블 설계, ERD작성자재/제품 재고 관리 - 자재/제품 재고 R품질관리 - 수입/공정/출하품질검사 CRU, 공정불량실적 R

이시연

**로그인,
비밀번호 재설정,
거래처·계약관리
권한관리,
접근제한**

로그인, 비밀번호 재설정

- Security를 활용하여 로그인
- 메일을 통해 일회용 인증코드를 전송하여 본인확인 후 비밀번호 변경

사원정보조회, 거래처관리, 계약관리, 권한 관리

- 사원정보조회, 거래처관리, 계약관리 검색창을 만들어 검색
- 권한관리를 통해 사원들의 권한을 변경

접근제한

- 사원의 권한에 따라 접근할 수 있는 페이지를 설정해 특정 페이지들을 접근하지 못하도록 제한
- 권한에 맞지 않는 페이지에 접근 시 커스텀 403 페이지를 설정

로그인

□view: 로그인 페이지

The image shows a login form with a logo for 'Cozy&Rest FURNITURE' featuring a chair and a sofa. The form has two input fields: 'Email' and 'Password'. Below the fields is a link '비밀번호를 잊으셨나요?'. At the bottom is a yellow '로그인' button.

이메일
Email

비밀번호
Password

비밀번호를 잊으셨나요?

로그인

SecurityConfig: 로그인 처리

```
.formLogin(form -> form
    .loginPage("/login")
    .loginProcessingUrl("/login")
    .usernameParameter("email")           // login 에 필요한 id
    .passwordParameter("password")        // login 에 필요한 password
    .failureHandler(FailureHandler())     /* 로그인 실패 핸들러 */
    .successHandler(SuccessHandler())      /* 로그인 성공 핸들러 */
    .defaultSuccessUrl("/", true)
    .permitAll()
```

Mapper: DB의 데이터와 비교하여 일치할 시에 로그인 처리

```
SELECT E_ID, E_NAME, E_EMAIL, E_PH, E_PWD, E_DP_NAME, E_ROLE
FROM EMPLOYEE
WHERE E_EMAIL = #{email}
```

Service: 로그인 시도 시 DB의 데이터와 비교하여 일치할 시에 로그인 처리

```
// DB에서 user 정보를 가져옴
MemberVO memberVO = memberMapper.selectByMember(email);
if (memberVO == null) return null;
```

로그인 성공처리

Config: 로그인 성공시 이전 페이지가 있을 경우 이전 페이지 반환

```
public class CustomAuthSuccessHandler extends SavedRequestAwareAuthenticationSuccessHandler {  
    public CustomAuthSuccessHandler(String defaultTargetUrl) { setDefaultTargetUrl(defaultTargetUrl); }  
  
    /**  
     * 인증에 성공할 경우 아래 매서드로 이동.  
     */  
    @Override no usages  
    public void onAuthenticationSuccess(HttpServletRequest request, HttpServletResponse response,  
                                         FilterChain chain, Authentication authentication)  
        throws IOException, ServletException {  
        HttpSession session = request.getSession();  
        if (session != null) {  
            String redirectUrl = (String) session.getAttribute("prevPage");  
            if (redirectUrl != null) {  
                session.removeAttribute("prevPage");  
                getRedirectStrategy().sendRedirect(request, response, redirectUrl);  
            } else {  
                super.onAuthenticationSuccess(request, response, authentication);  
            }  
        } else {  
            super.onAuthenticationSuccess(request, response, authentication);  
        }  
    }  
}
```

SecurityConfig: 로그인 성공 처리

```
@Bean 1 usage  
public CustomAuthSuccessHandler SuccessHandler() {  
    return new CustomAuthSuccessHandler(defaultTargetUrl: "/");  
}  
  
@Bean no usages  
public SecurityFilterChain securityFilterChain(HttpSecurity http) throws Exception {  
    http  
        .csrf(csrf -> csrf.disable())  
        .formLogin(form -> form  
            .loginPage("/login")  
            .loginProcessingUrl("/login")  
            .usernameParameter("email") // login 에 필요한 id  
            .passwordParameter("password") // login 에 필요한 password  
            .failureHandler(FailureHandler()) /* 로그인 실패 핸들러 */  
            .successHandler(SuccessHandler()) /* 로그인 성공 핸들러 */  
            .defaultSuccessUrl("/", true)  
        )  
        .permitAll()  
}
```

Controller: 로그인 처리

```
@GetMapping(path = {"/login"})  
public String login(@RequestParam(value = "error", required = false) String error,  
                    @RequestParam(value = "exception", required = false) String exception,  
                    Model model, HttpServletRequest request) {  
    // 이전페이지 URL 추출  
    String referrer = request.getHeader("Referer");  
    request.getSession().setAttribute("prevPage", referrer);  
    model.addAttribute(attributeName: "error", error);  
    model.addAttribute(attributeName: "exception", exception);  
    return "member/login";  
}
```

로그인 실패처리

□view: 로그인 실패처리

The login form for Cozy&Rest FURNITURE. It features a logo at the top left with the text "Cozy&Rest FURNITURE". Below the logo are two input fields: "이메일" (Email) and "비밀번호" (Password). A red rectangular box at the bottom contains the text "이메일과 비밀번호가 맞지 않습니다. 다시 확인해 주십시오" (Email and password do not match. Please check again). Below this box is a blue link "비밀번호를 잊으셨나요?" (Forgot password?). At the bottom right is a yellow button labeled "로그인" (Login).

□Config: 로그인 실패시 해당하는 오류에따라 메시지 전송

```
@Override 1 usage
public void onAuthenticationFailure(HttpServletRequest request, HttpServletResponse response,
                                     AuthenticationException exception)
throws IOException, ServletException {

    String errorMessage = null;
    if(exception instanceof BadCredentialsException || exception instanceof InternalAuthenticationServiceException){
        errorMessage = "이메일과 비밀번호가 맞지 않습니다. 다시 확인해 주십시오";
    } else if (exception instanceof UsernameNotFoundException) {
        errorMessage = "계정이 존재하지 않습니다. 관리자에게 문의하세요.";
    } else if (exception instanceof InternalAuthenticationServiceException) {
        errorMessage = "내부적으로 발생한 시스템 문제로 인해 요청을 처리할 수 없습니다. 관리자에게 문의하세요.";
    } else {
        errorMessage = "알 수 없는 이유로 로그인에 실패하였습니다. 관리자에게 문의하세요.";
    }
    errorMessage = URLDecoder.encode(errorMessage, enc: "UTF-8"); /* 한글 인코딩 깨진 문제 방지 */
    setDefaultFailureUrl("/login?error=true&exception="+errorMessage);
    super.onAuthenticationFailure(request, response, exception);
}
```

□JSP: Config의 오류 메시지를 받아서 view로 반환

```
<span class="">
    <c:if test="${error}">
        <p id="valid" class="alert alert-danger ">${exception}</p>
    </c:if>
</span>
```

비밀번호 변경

□view: 비밀번호 재설정 페이지

이메일
Email

인증번호
인증번호

발송

확인

□js - ajax: 비밀번호 재설정

```
// 인증번호 메일 발송 ajax
$("#email_auth_btn").click(function(){
    var email = $('#email').val();
    if(email == ''){
        alert("이메일을 입력해주세요.");
        return false;
    }

    $.ajax({
        type : "POST",
        url : "/emailAuth",
        data : {email : email},
        success: function(data){
            alert("인증번호가 발송되었습니다.");
            email_auth_cd = data;
        },
        error: function(data){
            alert("메일 발송에 실패했습니다.");
        }
    });
});
```

비밀번호 변경

□view: 비밀번호 재설정 페이지

The screenshot shows a password reset form. At the top is the C&R Furniture logo with the text "Cozy&Rest FURNITURE". Below it is a form with two input fields: "이메일" containing "lsm757504@naver.com" and "인증번호" (verification code). A red arrow points from the "발송" (Send) button to the corresponding Java code in the controller.

이메일
lsm757504@naver.com

인증번호
인증번호

발송

확인

★ C&R Furniture 프로젝트 비밀번호 변경 인증 이메일입니다.

보낸 사람: C&RFurniture@naver.com
받는 사람: lsm757504@naver.com
2024년 4월 22일 (월) 오후 12:54

C&R Furniture 프로젝트 비밀번호 변경 인증 이메일입니다.
인증 번호는 521358입니다.
해당 인증번호를 인증번호 확인란에 기입하여 주세요.

Controller: 로그인 페이지

```
@ResponseBody no usages
@RequestMapping(value = "/emailAuth", method = RequestMethod.POST)
public String emailAuth(String email) {
    log.info("전달 받은 이메일 주소 : " + email);
    int checkNum = RandomCode();
    sendEmail(email, checkNum);
    log.info("랜덤숫자 : " + checkNum);
    return Integer.toString(checkNum);}
```

□Controller: 랜덤 인증번호 6자리 생성

```
private int RandomCode() {
    //난수의 범위 111111 ~ 999999 (6자리 난수)
    Random random = new Random();
    return random.nextInt(bound: 888888) + 111111;}
```

□Controller: 메일 전송 세팅 함수

```
private void sendEmail(String email, int checkNum) {
    String setFrom = "C&RFurniture@naver.com";
    String title = "C&R Furniture 프로젝트 비밀번호 변경 인증 이메일입니다";
    String content = "C&R Furniture 프로젝트 비밀번호 변경 인증 이메일입니다. <br>" +
        "인증 번호는 " + checkNum + "입니다.<br>" +
        "해당 인증번호를 인증번호 확인란에 기입하여 주세요.;"
```

비밀번호 변경

□view: 비밀번호 재설정 페이지

이메일
lsm757504@naver.com

인증번호
554996

발송

확인

□js: 인증번호 유효성 검사,

```
// 인증번호 일치 검사
$("#code_ck").click(function(){
    if($('#email_auth_key').val() != email_auth_cd){
        alert("인증번호가 일치하지 않습니다.");
        return false;
    }
    if($('#email_auth_key').val() == email_auth_cd){
        alert("인증번호가 일치합니다.");
        $('.pw-area').css('visibility', 'visible');
        return false;
    }
});
```

인증번호
554996

새 비밀번호
Password

한번 더 입력
Retype Password

비밀번호 변경

사원정보조회, 거래처관리

□view: 사원정보조회

□ 사원정보조회

부서	사원명	부서	연락처	이메일
품질관리 2팀	임민지	품질관리 2팀	01087653921	qwert1234@naver.com
재고관리 1팀	최유진	재고관리 1팀	01087365912	asdfgh4567@gmail.com
생산 3팀	정현우	생산 3팀	01069217853	poiuyt3456@naver.com
품질관리 1팀	이수현	품질관리 1팀	01034892176	mnbvcx7890@naver.com

부서명	매니저	부서	연락처	이메일
생산 2팀	이수연	생산 2팀	01092631574	raccoon6310@naver.com
재고관리 1팀	김영수	재고관리 1팀	01054817693	mnb654@naver.com
생산 1팀	이지은	생산 1팀	01016398457	yhn987@naver.com
설비팀	박성민	설비팀	01075963812	wsx876@naver.com

□view: 거래처 관리

□ 거래처관리

거래처유형	발주/수주	거래처명	부서	연락처	이메일	주소
발주업체	나무마켓	나무마켓	발주업체	01057286913	Erandyn@dicelog.co.kr	서울특별시 강남구 역삼동 23-5번지
발주업체	원목킹	원목킹	발주업체	01098643157	Jeraineal@dicelog.co.kr	인천광역시 남동구 만수동 14-7번지
발주업체	소나무가게	소나무가게	발주업체	01043768291	Padez@dicelog.co.kr	대전광역시 서구 둔산동 58-3번지

□domain: 사원정보 검색을 위한 클래스

```
private String find_DP_name; // 부서 선택  
private String find_emp_name; // 사원이름 검색
```

□Mapper.xml: 사원정보조회 검색

```
<!-- 부서명 검색 -->  
<if test="find_DP_name != null and find_DP_name != ''"  
AND E_DP_NAME Like '%' || #{find_DP_name} || '%'>  
</if>  
<!-- 사원명 검색 -->  
<if test="find_emp_name != null and find_emp_name != ''"  
AND E_NAME Like '%' || #{find_emp_name} || '%'>  
</if>
```

□domain: 거래처 검색을 위한 클래스

```
private String find_c_div; // 거래처 구분 선택  
private String find_c_name; // 거래처명 선택
```

□Mapper.xml: 거래처관리 검색

```
<!-- 거래처유형 검색 -->  
<if test="find_c_div != null and find_c_div != ''"  
AND C_DIV like '%' || #{find_c_div} || '%'>  
</if>  
<!-- 거래처명 검색 -->  
<if test="find_c_name != null and find_c_name != ''"  
AND C_NAME like '%' || #{find_c_name} || '%'>  
</if>
```

계약관리

□view: 계약관리

계약관리

거래처명	<input type="text"/>		자재명/번호	<input type="text"/>		입고일	연도. 월. 일. <input type="text"/> ~ 연도. 월. 일. <input type="text"/>			
계약날짜	연도. 월. 일. <input type="text"/>	연도. 월. 일. <input type="text"/>	제품명/번호	<input type="text"/>		출고일	연도. 월. 일. <input type="text"/> ~ 연도. 월. 일. <input type="text"/>			
계약목록										
NO	구분	거래처명	자재	제품	금액	수량	수량단위	계약날짜	입고일	출고일
1000001	발주	나무마켓	소나무		4500000	10	m	2024-04-04	2024-04-11	
1000002	발주	원목킹	참나무		2500000	10	m	2024-04-05	2024-04-11	
1000003	발주	소나무가게	자작나무		3000000	10	m	2024-04-06	2024-04-11	
1000004	발주	원목스토어	너도밤나무		2000000	10	m	2024-04-07	2024-04-11	

□domain: 계약관리 검색을 위한 클래스

```

@Getter 8 usages
@Setter
public class ContractSearch {
    private String find_item; // 제품명/번호 검색
    private String find_material; // 자재명/번호 검색
    private String find_name; // 거래처명 검색
    private String find_contract_start_date; // 계약날짜 검색 1
    private String find_contract_end_date; // 계약날짜 검색 2
    private String find_ib_start_Date; // 입고일 검색 1
    private String find_ib_end_Date; // 입고일 검색 2
    private String find_ob_start_Date; // 출고일 검색 1
    private String find_ob_end_Date; // 출고일 검색 2
}

```

□Mapper.xml: 계약관리 검색

```


<if test="find_name != null and find_name != ''">
    AND cp.C_NAME LIKE '%' || #{find_name} || '%'
</if>

<if test="find_contract_start_date != null and find_contract_start_date != '' and find_contract_end_date != null and find_contract_end_date != ''">
    AND c.CT_DATE BETWEEN TO_DATE(#{find_contract_start_date}, 'yyyy-mm-dd')
    AND TO_DATE(#{find_contract_end_date}, 'yyyy-mm-dd')
</if>

<if test="find_ib_start_Date != null and find_ib_start_Date != '' and find_ib_end_Date != null and find_ib_end_Date != ''">
    AND c.CT_IB_DATE BETWEEN TO_DATE(#{find_ib_start_Date}, 'yyyy-mm-dd')
    AND TO_DATE(#{find_ib_end_Date}, 'yyyy-mm-dd')
</if>

<if test="find_ob_start_Date != null and find_ob_start_Date != '' and find_ob_end_Date != null and find_ob_end_Date != ''">
    AND c.CT_OB_DATE BETWEEN TO_DATE(#{find_ob_start_Date}, 'yyyy-mm-dd')
    AND TO_DATE(#{find_ob_end_Date}, 'yyyy-mm-dd')
</if>

<if test="find_item != null and find_item != ''">
    AND (c.CT_ITEM_ID LIKE '%' || #{find_item} || '%' OR i.I_NAME like '%' || #{find_item} || '%')
</if>

<if test="find_material != null and find_material != ''">
    AND (c.CT_MATERIAL_ID LIKE '%' || #{find_material} || '%' OR m.M_NAME like '%' || #{find_material} || '%')
</if>

```

권한관리 - 권한 변경

□view: 사용자별 권한 관리

□ 사용자별 권한 관리

	사원번호	사원명	권한	부서	연락처	이메일
<input type="checkbox"/>	100128	박지우	사원	생산 3팀	01011110047	qwldn5@example.com
<input type="checkbox"/>	100129	김가온	사원	생산 1팀	01011110048	rjdh5742@naver.com
<input type="checkbox"/>	100130	정준서	사원	품질관리 1팀	01011110049	wwnstj84@example.com
<input type="checkbox"/>	100131	최하율	사원	생산 3팀	01011110050	wgkdbf84@naver.com
<input type="checkbox"/>	100132	이수민	사원	설비팀	01011110051	dtnals78@naver.com
<input type="checkbox"/>	100133	김하윤	사원	생산 2팀	01011110052	rgkdus02@example.com
<input type="checkbox"/>	100134	박주원	사원	생산 1팀	01011110053	qwndnjs62@example.com
<input type="checkbox"/>	100135	정다은	사원	생산 1팀	01011110054	wjejdns64@naver.com
<input checked="" type="checkbox"/>	100136	최시윤	사원	생산 1팀	01011110055	tcldbhs87@naver.com
<input checked="" type="checkbox"/>	100137	홍수빈	사원	품질관리 2팀	01011110056	gtnals62@naver.com
<input checked="" type="checkbox"/>	100138	이현서	사원	기획팀	01054651234	qkqh123@naver.com

□js: 권한 변경 처리

```
/** 권한변경 */
function checkboxArr() {
    // 배열 선언
    var checkArr = [];
    if($('.list-checkBox:checked').length<1) {
        alert('권한 변경할 사원을 체크해주세요');
    } else {
        //선언한 배열에 데이터를 삽입
        $(".list-checkBox:checked").each(function(i) {
            // 선택된 체크박스를 순회하면서 데이터 추출
            var row = $(this).closest('tr'); // 현재 체크박스가 속한 행 선택
            var id = row.find("#m_id").text(); // 행에서 id 값을 가져옴
            var name = row.find("#m_name").text(); // 행에서 이름 값을 가져옴
            var role = row.find("#m_role").text(); // 행에서 역할 값을 가져옴
            // 데이터를 객체로 만들어 배열에 추가
            var param = { id: id, name: name, role: role };
            checkArr.push(param);
        })
        // 권한 변경 확인 체크
        if(confirm('사원권한 변경을 하시겠습니까?')) {
            // ajax를 통하여 전송
            $.ajax({
                type : "POST",
                url : "/D/modifyMemberList",
                contentType: "application/json",
                data : JSON.stringify(checkArr),
                dataType: "text",
                success: function(text){
                    alert("권한 변경이 완료되었습니다.");
                    location.href="/M/member/memberRole";
                },
                error: function(request, status, error, xhr){
                    alert("권한 변경을 실패했습니다. \n 에러코드: "+xhr.status);
                }
            });
        } else {
            return false;
        }
    }
}
```

□Controller: 권한 수정 - ajax 처리

```
@ResponseBody no usages
@RequestMapping(value = "/D/modifyMemberList", method = RequestMethod.POST)
public String modifyMemberList(@RequestBody List<MemberVO> checkList) {
    for (MemberVO member:checkList) { memberService.modifyRole(member); }
    return "success";
}
```

권한관리 - 권한 변경

□view: 사용자별 권한 관리

□ 사용자별 권한 관리

The screenshot shows a table with columns: 사원번호 (Employee ID), 사원명 (Employee Name), 권한 (Permission), 부서 (Department), 연락처 (Contact), and 이메일 (Email). There are checkboxes in the first column for selecting rows. A red box highlights the '사원등록' button at the top right of the page.

	사원번호	사원명	권한	부서	연락처	이메일
<input type="checkbox"/>	100128	박지우	사원	생산 3팀	01011110047	qwldn5@example.com
<input type="checkbox"/>	100129	김가온	사원	생산 1팀	01011110048	rjdhs742@naver.com
<input type="checkbox"/>	100130	정준서	사원	품질관리 1팀	01011110049	wwnstj84@example.com
<input type="checkbox"/>	100131	최하율	사원	생산 3팀	01011110050	wgkdbf84@naver.com
<input type="checkbox"/>	100132	이수민	사원	설비팀	01011110051	dtnals78@naver.com
<input type="checkbox"/>	100133	김하윤	사원	생산 2팀	01011110052	rgkdus02@example.com
<input type="checkbox"/>	100134	박주원	사원	생산 1팀	01011110053	qwndnjs62@example.com
<input type="checkbox"/>	100135	정다은	사원	생산 1팀	01011110054	wjejdns64@naver.com
<input checked="" type="checkbox"/>	100136	최시윤	사원	생산 1팀	01011110055	tcldbhs87@naver.com
<input checked="" type="checkbox"/>	100137	홍수빈	사원	품질관리 2팀	01011110056	gtndls02@naver.com
<input checked="" type="checkbox"/>	100138	이현서	사원	기획팀	01054651234	qkqh123@naver.com

□view: 사원등록 모달창

The modal window has fields for '사원명' (Employee Name), '부서 선택' (Department Selection), '연락처' (Contact), and 'email'. At the bottom are '등록' (Register) and '취소' (Cancel) buttons. A red arrow points from the '사원등록' button in the main view to this modal.

□Controller: □사원등록

```
@PostMapping("/addStaffMember") no usages
public String addStaffMember(MemberVO memberVO, RedirectAttributes rttr) {
    int rtn = memberService.addStaff(memberVO);
    rttr.addFlashAttribute( attributeName: "staffInsertSuccessCount", rtn);
    return "redirect:/memberRole";
}
```

□Mapper: 사원 등록

```
INSERT INTO EMPLOYEE
VALUES ( #{name}, #{email}, #{ph}, #{dp_name} )
```

느낀 점(이시연)

[좋았던 점]

- 프로젝트 설계를 탄탄하게 하여, 프로젝트에 대한 이해도를 높이고 개발을 원활하게 진행 할 수 있어서 좋았음.
- 피그마로 화면을 설계하고 개발을 시작하니 차근차근 따라가면서 진행 할 수 있어서 좋았음.
- 팀원들과 소통이 원활하여 서로 어려운 문제가 생기면 바로 도와줄 수 있어서 좋았음.

[아쉬웠던 점]

- 개발 기간이 조금 더 길었으면 좀 더 다양한 방법을 시도해볼 수 있었을 거 같은데 짧아서 아쉬웠음.
- 접근 제한 처리가 조금 부족한 거 같아서 아쉬웠음.
- 권한 제한으로 인한 접근 불가 페이지를 다른 방법을 써서 하고 싶었지만 그러지 못해서 아쉬웠음.

—
이현주

BOM 관리, 자재/제품정보관리, 작업자스케줄조회

BOM관리

- 제품 번호에 해당하는 제품 리스트 조회
- BOM 관리에 등록된 자재 수량만 수정
- BOM 추가 후 자재 단위 및 수량을 바로 수정 가능하게 함
- BOM 등록에 제품 및 자재 관리 버튼 추가
- BOM 등록에서 제품 자재 검색

제품관리

- 제품명, 제품번호, 제품용도 검색
- 자재 - 조회, 등록, 수정

자재관리

- 자재명, 자재번호, 자재용도 검색
- 자재 - 조회, 등록, 수정

작업자관리

- 전체 사원 조회 및 검색
- 관리자 또는 매니저가 사원 스케줄을 월별 확인

자재 & 제품 정보 관리: 검색 인터페이스 & 조회

VIEW

자재 정보 관리

--선택--

--선택--

자재번호
자재명
자재용도

자재 번호

자재명

자재 용도

검색

자재등록

DTO: 삼항 연산자를 이용해 배열을 초기화하고 값을 할당

```
@Getter  
@Setter  
public class ItemInfoSearch {  
  
    private String type;  
    private String keyword;  
  
    no usages - luna  
    public String[] getTypeArr() { return type == null ? new String[] {} : new String[] {type}; }  
}
```

JSP: name, value, <c:out>을 통해 서버와 연결해 값을 전달

```
<select name='type'>  
    <option value='--선택--'>--선택--</option>  
    <option value="INum"><c:out value="${itemSearch.type == 'INum' ? 'selected' : ''}" />제품번호</option>  
    <option value="IName"><c:out value="${itemSearch.type == 'IName' ? 'selected' : ''}" />제품명</option>  
    <option value="IUses"><c:out value="${itemSearch.type == 'IUses' ? 'selected' : ''}" />제품용도</option>  
</select>  
</div>  
<div class="col-sm-1">  
    <input type="text" name='keyword' value='<c:out value="${itemSearch.keyword}" />' />
```

JavaScript:

e.preventDefault()는 <button> 요소의 클릭 이벤트 기본 동작인 form 제출 방지 후 submit 실행

```
let searchItemForm = $("#searchItemForm");  
$("#searchItemForm button").on("click",  
    function(e) {  
        if (!searchItemForm.find("option:selected").val()) {  
            alert("검색종류를 선택하세요");  
            return false;  
        }  
        if (!searchItemForm.find("input[name='keyword']").val()) {  
            alert("키워드를 입력하세요");  
            return false;  
        }  
        e.preventDefault();  
  
        searchItemForm.submit();  
   });
```

Mapper.xml: 검색어가 있을 경우 <foreach>,<if> 구문 실행

```
<select id="getItemInfoList" resultType="com.cnr_furniture.domain.ItemInfo.ItemInfoVO">  
    SELECT ROWNUM rn, I_ID, I_NAME, I_COLOR, I_STANDARD, I_TYPE, IUSES  
    FROM ITEM  
    WHERE 1=1  
    <foreach item="type" collection="typeArr">  
        <if test="type == 'INum'.toString()">  
            and I_ID like '%' || #{keyword} || '%'  
        </if>  
        <if test="type == 'IName'.toString()">  
            and I_NAME like '%' || #{keyword} || '%'  
        </if>  
        <if test="type == 'IUses'.toString()">  
            and IUSES like '%' || #{keyword} || '%'  
        </if>  
    </foreach>  
    ORDER BY I_ID ASC  
</select>
```

자재 & 제품 정보 관리: 등록

VIEW: 제품등록 클릭 시 모달창

제품 정보 관리

--선택--

검색

제품등록

제품정보 등록

제품명

제품 색상

제품 규격

제품 종류

제품 용도

필수! 반제품 또는 완제품으로 입력해주세요

등록 취소

JSP: 제품등록 클릭시 부트스트랩 모달창 pop-up

```
<div class="modal fade" id="myModal1" tabindex="-1" role="dialog" aria-labelledby="myModalLabel1">
<div class="modal-dialog" role="document">
<div class="modal-content">
<div class="modal-header">
<h4 class="mt-modal-title" id="myModalLabel1">제품정보 등록</h4>
</div>
<div class="modal-body">
<form action=".//itemInsert" method="post" id="insertItemForm" onSubmit="return false">
<label for="modalUpdateName" class="form-label">제품명</label>
<input type="text" name="i_name" id="i_name" class="form-control" autocomplete="off">

<label for="modalProductAmount" class="form-label">제품 색상</label>
<input type="text" name="i_color" id="i_color" class="form-control" autocomplete="off">

<label for="modalProductAmount" class="form-label">제품 규격</label>
<input type="text" name="i_standard" id="i_standard" class="form-control" autocomplete="off">

<label for="modalProductAmount" class="form-label">제품 종류</label>
<input type="text" name="i_type" id="i_type" class="form-control" autocomplete="off">

<label for="modalProductAmount" class="form-label">제품 용도</label>
<input type="text" name="i_uses" id="i_uses" class="form-control" autocomplete="off" placeholder="필수! 반제품 또는 완제품으로 입력해주세요">
</form>
</div>
<div class="modal-footer">
<button type="submit" class="btn btn-primary" onClick="insertItemBox()>등록</button>
<button type="button" class="btn btn-danger" data-dismiss="modal">취소</button>
</div>
</div>
</div>
```

자재 & 제품 정보 관리: 수정

VIEW: 제품번호 클릭 시 해당 제품정보 불러오기

제품 목록							
NO	제품 번호	제품명	제품 색상	제품 규격	제품 종류	제품 용도	수정
1	10000001	의자-A	무색	성인용	의자	완제품	<button>수정</button>
2	10000002	의자-B	베이지	성인용	의자	완제품	<button>수정</button>

MODAL

제품정보 수정

제품 번호	제품명
10000001	의자-A
제품 색상	제품 규격
무색	성인용
제품 종류	제품 용도
의자	완제품

수정 **취소**

jQuery: 선택된 값을 jQuery를 사용하여 AJAX 비동기식 호출

```
function ItemUpdates(i_id) {
    $.get("/itemUpdate/" + i_id, function(result) {
        console.log("itemUpdate:", result);

        $('#i_id2').val(result.i_id);
        $('#i_name2').val(result.i_name);
        $('#i_color2').val(result.i_color);
        $('#i_standard2').val(result.i_standard);
        $('#i_type2').val(result.i_type);
        $('#i_uses2').val(result.i_uses);

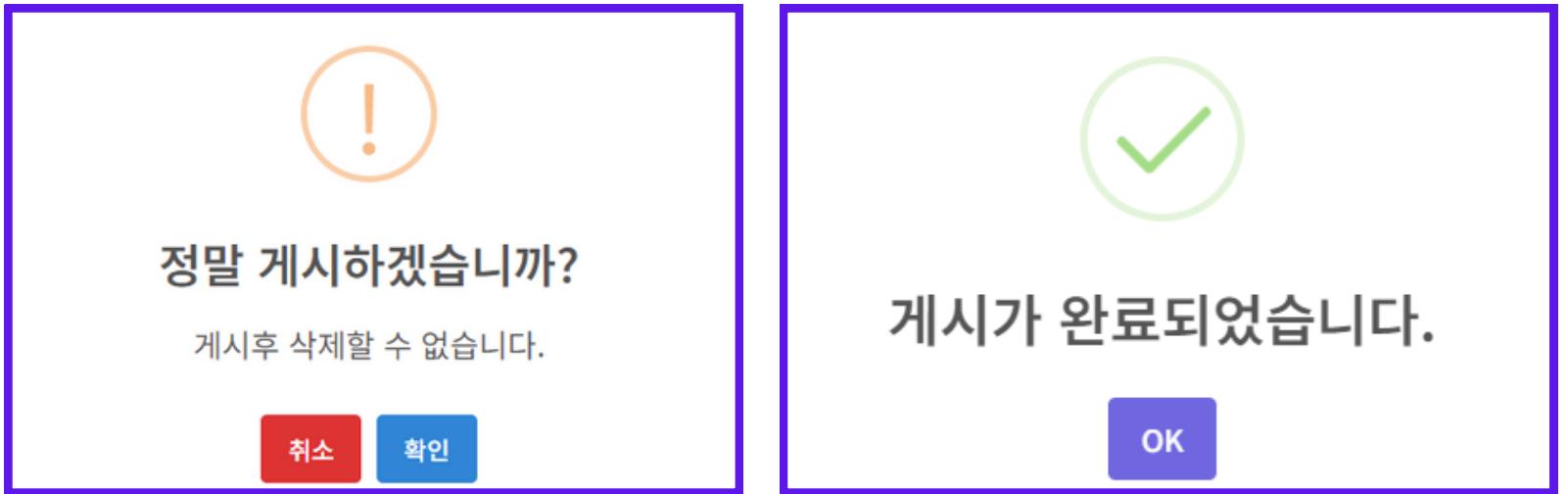
    }).fail(function(xhr, status, err){
        console.log("itemUpdate err:", err);
    });
}
```

Controller: 수정 할 값을 AJAX로 호출하는 controller method

```
@ResponseBody
@GetMapping(value = "/itemUpdate/{i_id}", produces = MediaType.APPLICATION_JSON_VALUE)
public ResponseEntity<ItemInfoVO> get(@PathVariable("i_id") int i_id) {
    return new ResponseEntity<>(itemInfoService.getOneItem(i_id), HttpStatus.OK);
}
```

자재 & 제품 정보 관리: 등록 및 sweetAlert2

VIEW



JSP: 외부 라이브러리 등록

```
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/sweetalert2@11.4.10/dist/sweetalert2.min.css">
<script src="https://cdn.jsdelivr.net/npm/sweetalert2@11.4.10/dist/sweetalert2.min.js"></script>
<script src="https://cdn.jsdelivr.net/npm/sweetalert2@9"></script>
```

jQuery: 모든 필드 값 확인 true/false

```
/*제품 등록시 사용 */
function checkAllFields() {
    return $('#i_name').val() && $('#i_color').val()
        && $('#i_standard').val() && $('#i_type').val()
        && $('#i_uses').val();
}
```



JavaScript: form 제출과 sweetAlert2 적용

```
function insertItemBox() {
    console.log($('#i_name').val());

    if (!checkAllFields()) {
        Swal.fire({
            title: '입력 오류',
            text: '모든 정보를 입력하세요.',
            icon: 'error',
            confirmButtonColor: '#48088A'
        });
        return;
    }

    // 사용자가 등록을 원하는지 확인하는 창 표시
    Swal.fire({
        title: '정말 게시하겠습니까?',
        text: '게시후 삭제할 수 없습니다.',
        icon: 'warning',
        showCancelButton: true, // cancel버튼 보이기. 기본은 원래 없음
        confirmButtonColor: '#3085d6', // confirm 버튼 색깔 지정
        cancelButtonColor: '#d33', // cancel 버튼 색깔 지정
        confirmButtonText: '확인', // confirm 버튼 텍스트 지정
        cancelButtonText: '취소', // cancel 버튼 텍스트 지정
        reverseButtons: true, // 버튼 순서 거꾸로
    }).then(result => {
        if (result.isConfirmed) {
            Swal.fire('게시가 완료되었습니다.', '', 'success');
            setTimeout(function() {
                document.getElementById('insertItemForm').submit();
            }, 2000); // 2초 후에 폼 제출
        } else if (result.isDismissed) { // 만약 모달창에서 cancel 버튼을 눌렀다면
            Swal.fire('게시가 취소되었습니다.', '', 'info');
        }
    })
}
```

BOM 관리: 세부목록 조회

VIEW: 제품번호 클릭 시 해당 자재리스트 출력

jQuery: ajax를 사용해 제품(BOM) 목록을 화면에 동적으로 표시

```
let bomListUL = $(".checkBomList");
let tempQuantity = 0;

function showBomList(i_id) {
    // i_id를 가져와서 화면에 뿌려주는 함수 선언 i_id=B_ITEM_ID
    $.get("/bomList/" + i_id, function(result) {
        // 1. Bom 목록 rest ajax로 가져오기, ajax 함수 콜 성공시 처리
        console.log("showBomList:", result);

        var str = "";
        if (!result || result.length === 0) {
            bomListUL.html(""); // 제품에 등록된 bom이 없을 때 비우기
            return;
        }
    });
}
```

```
for (var i = 0; i < result.length; i++) {
    str += "<tr>";
    str += "    <td>" + result[i].rn + "</td>";
    str += "    <td>" + result[i].b_material_id + "</td>";
    str += "    <td>" + result[i].m_name + "</td>";
    str += "    <td>" + result[i].b_unit + "</td>";
    str += "    <td id='mQuantity-" + result[i].b_material_id + "'>" + result[i].b_material_quantity + "</td>";
    str += "    <td id='mQuantity-backup-' + result[i].b_material_id + '' style='display: none;'>" + result[i].b_material_quantity + "</td>";
    str += "    <td>";
    str += "        <div class='modifyBom' id='modifyBom-' + result[i].b_material_id + '' data-bmid='" + result[i].b_material_id + "'";
    str += "            data-itemid='" + result[i].b_item_id + "'>수정</div>";
    str += "    </td>";
    str += "</tr>";
}

bomListUL.html(str); // 결과를 HTML에 삽입(html 렌더링한다)
```

제품 목록			BOM 세부목록			
NO	제품 번호	제품명	NO	자재 번호	자재명	단위
1	10000001	의자-A	1	20000001	소나무	m
2	10000002	의자-B				

```
<td onClick='showBomList(<c:out value="${Item.i_id}" />)' class="itemNumBtn">
    <c:out value="${Item.i_id}" />
```

BOM 관리: 세부목록 설정

jQuery: ajax를 사용해 제품(BOM) 수량을 화면에 동적으로 수정

```
let bomListUL = $(".checkBomList");
let tempQuantity = 0;
```

1. 수정할 값을 replaceWith로 변경

```
$('.modifyBom').on('click', function(e) {
    var bmId = $(this).attr('data-bmid'); // BOM 자재ID 가져오기
    var itemId = $(this).attr('data-itemid'); // BOM 제품ID 가져오기

    const mQuantity = $('#mQuantity-' + bmId).text(); // 실제 내용 가져오기
    console.log('modfiyBom', mQuantity);
    tempQuantity = mQuantity;

    $('#mQuantity-' + bmId).replaceWith("<input type='text' id='mQuantity-" + bmId + "' value='" + (mQuantity || $('#mQuantity-' + bmId).val()) + "' />");
```

2. ajax 실제 수정 처리

```
$.ajax({
    // request 처리
    type : 'post', // form의 method속성 값
    url : '/bom/' + bomMaterialId + "/" + bItemId, // form의 action값
    //url : '/bom/' + bomMaterialId, // form의 action값
    data : JSON.stringify(bom), // json으로 string처리하면서 파라미터 전달
    contentType : "application/json; charset=utf-8", // content-type지정
    // response 처리
    success : function(result, status, xhr) { // call 성공시 오는 처리되는 함수
        $('#mQuantity-backup-' + bomMaterialId).text($('#mQuantity-' + bomMaterialId).val());
```

3. 수정 완료 후 tempQuantity를 이용해 화면처리

```
if (tempQuantity != $('#mQuantity-' + bomMaterialId).val()) {
    $('#mQuantity-' + bomMaterialId).replaceWith("<td id='mQuantity-" + bomMaterialId + "'>" + $('#mQuantity-' + bomMaterialId).val() + "</td>");
```

VIEW: 수정 전

BOM 세부목록					
NO	자재 번호	자재명	단위	수량	수정
1	20000001	소나무	m	1	수정

VIEW: 수정 후

NO	자재 번호	자재명	단위	수량	수정
1	20000001	소나무	m	2	수정

BOM 등록: 자재 추가

VIEW

jQuery: form 내부의 값 가져오기

```
var itemId = $('#itemId').val();
var mtId = $('#mtId').val();
var mtUnit = $('#mtUnit').val();
var mtQuantity = $('#mtQuantity').val();
```

AJAX: 등록 후 성공시 새로운 행을 출력

```
$.ajax({
    // request 처리
    type : 'post', // form의 method 속성 값
    url : '/bom/insert/' + itemId + "/" + mtId + "/" + mtUnit + "/" + mtQuantity, // form의
    data: JSON.stringify({ itemId: itemId, mtId: mtId, mtUnit: mtUnit, mtQuantity: mtQuantity }),
    contentType : "application/json; charset=utf-8", // content-type 지정
    // response 처리
    success : function(result, status, xhr) { // call 성공시 오는 처리되는 함수
        // 등록 성공 시
        Swal.fire('Bom 등록이 완료되었습니다.', '', 'success');

        // 새로운 행(tr)을 생성하여 HTML 문자열로 저장
        var newRow =
            "<tr>" +
            "<td class='itemId'>" + itemId + "</td>" +
            "<td class='mtId'>" + mtId + "</td>" +
            "<td class='mtUnit'>" + mtUnit + "</td>" +
            "<td class='mtQuantity'>" + mtQuantity + "</td>" +
            "<td class='modifyBomInserted'>수정</td>" +
            "</tr>";

        // 생성한 행을 테이블에 추가
        $('.addBomListUL').append(newRow);
        /* alert('BOM이 등록되었습니다'); */
    }
});
```

BOM 등록

제품 번호	10000020	자재 번호	20000004	
자재 단위	test	자재 수량	1	
<input type="button" value="추가"/>				
제품 번호	자재 번호	단위	수량	수정
10000020	20000001	test	1	수정

Controller: POST 요청을 처리와 ajax 사용해 호출

```
@RequestMapping(method = { RequestMethod.POST })
, value = "/bom/insert/{b_item_id}/{b_material_id}/{b_unit}/{b_material_quantity}"
)
public ResponseEntity<String> bomInsert(
    @RequestBody BomVO bomVO,
    @PathVariable("b_item_id") int b_item_id,
    @PathVariable("b_material_id") int b_material_id,
    @PathVariable("b_unit") String b_unit,
    @PathVariable("b_material_quantity") int b_material_quantity
){
    bomVO.setB_item_id(b_item_id);
    bomVO.setB_material_id(b_material_id);
    bomVO.setB_unit(b_unit);
    bomVO.setB_material_quantity(b_material_quantity);

    int insertSuccess = bomService.insertBomList(bomVO);

    return insertSuccess == 1
        ? new ResponseEntity<>(@body: "success", HttpStatus.OK)
        : new ResponseEntity<>(HttpStatus.INTERNAL_SERVER_ERROR);
}
```

작업자스케줄조회: 월별 스케줄러

VIEW

The screenshot shows a monthly calendar for April 2024. The sidebar lists employees with their shift details. Several events are highlighted in red boxes, including '국회의원선거일' on April 10th and '노동절' on April 1st.

사원번호	사원명	부서명
100001	김다은	설비팀
100002	이시연	재고관리 2팀
100003	이현주	기획팀
100004	조다혜	기획팀
100005	정수하	품질관리 1팀
100006	배지현	생산 2팀
100007	이수연	생산 2팀
100008	김영수	재고관리 1팀
100009	이지은	생산 1팀
100010	박성민	설비팀
100011	최미경	재고관리 2팀
100012	전진호	생산 2팀

JSP: fullcalendar5.8 외부 라이브러리 등록

```
<link
  href='https://cdn.jsdelivr.net/npm/fullcalendar@5.8.0/main.min.css'
  rel='stylesheet' />
<script
  src='https://cdn.jsdelivr.net/npm/fullcalendar@5.8.0/main.min.js'></script>
<!-- fullcalendar 언어 CDN --&gt;
&lt;script
  src='https://cdn.jsdelivr.net/npm/fullcalendar@5.8.0/locales-all.min.js'&gt;&lt;/script&gt;</pre>
```

fullcalendar function: events에서 구글캘린더 API 적용, ajax 실행

```
locale: 'ko', // 한국어 설정
googleCalendarApiKey: 'google api id',
eventSources: [
  {
    googleCalendarId: 'ko.south_korea#holiday@google.com',
    className: 'ko_event',
    color: 'red',
    //textColor: 'black'
  },
  function(fetchInfo, successCallback, failureCallback) {
    console.log("들어왔다");

    // AJAX 요청
    $.ajax({
      type: "GET",
      url: "/schedule",
      contentType : "application/json; charset=utf-8", // content-type
      success: function(response) {
        console.log("success 들어왔다");
        var events = [];
        if (response != null) {
          var result = response;
          // 서버에서 받은 데이터로 이벤트 배열 구성
          for (var i = 0; i < result.length; i++) {
            events.push({
              title: result[i].name,
              start: result[i].start,
              end: result[i].end
            });
          }
          // 이벤트 배열을 FullCalendar에 전달하여 캘린더에 표시
          successCallback(events);
        } else {
          // 서버 응답 오류 처리
          failureCallback('Invalid response from server');
        }
      },
      error: function(xhr, status, error) {
        // AJAX 오류 처리
      }
    });
  }
],
```



느낀 점(이현주)

[좋았던 점]

- 설계부터 시작해 마지막 작업까지 모두 책임감을 갖고 프로젝트를 마친 점.
- 서로에 대한 존중과 신뢰 기반의 팀워크.
- 일정에 맞춰 진행한 점.
- 6개월동안 배운 과정을 활용한 점.
- 깃, 깃허브 사용으로 팀원들과 빠른 코드공유, 에러확인 및 해결한 점.

[아쉬웠던 점]

- fullcalendar 응용해 사원번호 클릭시 해당사원의 스케줄 표시를 구현하지 못한 점이 아쉬웠음.
- BOM 등록 시 두 개 이상을 한 번에 추가 구현 못한 점이 아쉬웠음.
- MES에 필요한 제품자재 바코드 또는 POP를 구현 못한 점이 아쉬웠음.

—
조다혜

제조지시, 제조수행지시, 공정정보관리

제조지시

- 제조 지시 조회
- 제조지시 선택:
 - ✓ 해당 제조 지시의 계약 내역이 동적으로 화면에 표시
- 계약 내역:
 - ✓ 라이브러리를 사용해 PDF 형태로 다운
- 제조지시등록:
 - ✓ 원하는 계약 선택후 플러스 버튼을 클릭 -> 계약번호, 제품 번호, 제품 수량 (*1.2배) 자동으로 입력.

제조수행지시 공정정보관리

- 제조LOT번호, 지시일자, 공정번호로 조회
- 제조 LOT 번호를 선택:
 - ✓ LOT에 등록된 제조 지시 번호만을 표시하며, 이를 선택할 때 관련 제품 번호와 공정 번호가 자동으로 동기화 되어 표시

공정정보관리

- 설비 조회 및 조회 리셋 기능(모달창)
- 설비를선택:
 - ✓ 설비가 추가된 설비 목록에 동적으로 업데이트
- 추가된 설비 목록:
 - ✓ 해당되는 번호 체크 후 버튼을 클릭->해당 설비 번호가 동적으로 입력

검색-조회

VIEW

제조계획지시 조회

계획일자 연도-월-일 ~ 연도-월-일 제품번호 조회

제조수행지시 조회

제조LOT번호 지시일자 연도-월-일 ~ 연도-월-일 공정번호 조회

공정정보관리

공정번호 조회

JSP 파일

```
<select class="process-select" id="proPi" name="p_pi_id">
    <option value="">--선택--</option>
    <c:forEach items="${insPiList}" var="proLot">
        <option value="${proLot.ins_pi_id}" ${proLot.ins_pi_id == processDate.ins_pi_id ? 'selected' : ''}>${proLot.ins_pi_id}</option>
    </c:forEach>
</select>

<div class="proName">
    <div class="searchProbar">제조LOT번호</div>
    <input list="proRunList" name="pLotId" class="processBox" aria-label=".form-select-sm example" value="${param.pLotId}">
        <datalist id="proRunList">
            <option value="--선택--">--선택--</option>
            <c:forEach items="${proRunList}" var="pLotId">
                <option value="${pLotId.p_lot_id}">${pLotId.p_lot_id}</option>
            </c:forEach>
        </datalist>
</div>
```

10000001	1001
10000001 의자-A	1001 원자재준비
10000002	1002
10000002 의자-B	1002 오버레이
10000003	1003
10000003 의자-C	1003 재단
10000004	1004
10000004 의자-D	1004 에지 마감 공정
10000005	1005
10000005 의자-E	1005 NC작업
10000006	1006
10000006 침대-A	1006 보링작업
10000007	1007
10000007 침대-B	1007 보링작업
10000008	1008
10000008 침대-C	1008 조립
10000009	1009
10000009 침대-D	1009 조립
10000010	1010
10000010 침대-E	1010 포장
10000011	1011
10000011 소파-A	1011 적재
10000012	1012
10000012 소파-B	1012 원자재준비
10000013	1013
	1013 가재단(roughing cut)
	1014
	1014 1차 면적(cutting)

제조지시-등록

제조지시등록

VIEW

담당사원번호	<input type="text"/>	제품번호	<input type="text" value="10000001"/>	제조LOT번호	<input type="text"/>
계약번호	<input type="text" value="1000013"/>	공정번호	--선택--	계획수량	<input type="text" value="4800"/>
계획착수일	연도-월-일	계획완수일	연도-월-일		
<input type="button" value="추가"/> <input type="button" value="등록"/>					
담당사원번호	제품번호	제조LOT번호	계약번호	공정번호	계획수량
계약착수일	계약완수일				
<input type="button" value="삭제"/>					

제조지시

계획일자

연도-월-일

제품번호

조회

제조시시속록

선택 내보내

	제조LOT번호	계약번호	계획착수일	계획완수일	제품번호	계획수량
<input type="checkbox"/>	300001	1000013	2024-04-02	2024-04-09	10000001	4800
<input type="checkbox"/>	300002	1000017	2024-04-01	2024-04-08	10000006	3600
<input type="checkbox"/>	300003	1000021	2024-04-02	2024-04-09	10000011	4800
<input type="checkbox"/>	300004	1000024	2024-04-01	2024-04-07	10000016	4800
<input type="checkbox"/>	300005	1000025	2024-04-02	2024-04-13	10000021	6000

제조지시에서 등록 모달창에 데이터 추출 함수

연도-월-일 ~ 연도-월-일 조회

+ [+]

계약목록

NO	계약번호	제품번호	계약수량	계약체결일	계약출고일
<input checked="" type="checkbox"/>	1000013	10000001	4000	2024-04-02	2024-04-09
<input type="checkbox"/>	1000014	10000002	300	2024-04-15	2024-04-22
<input type="checkbox"/>	1000015	10000003	300	2024-04-15	2024-04-22

취소

JavaScript

```
// 제조지시
// 제조지시에서 등록 모달창에 데이터 추출 함수
$(document).ready(function() {
    $('#addText').click(function() {
        $('#ctProList input:checked').each(function() {
            var row = $(this).closest('tr');
            var ct_id = row.find('td:nth-child(2)').text().trim(); // 계약번호 추출
            var ct_item_id = row.find('td:nth-child(3)').text().trim(); // 제품번호 추출
            var ct_quantity = parseFloat(row.find('td:nth-child(4)').text().trim()); // 수량 추출

            var calculated_quantity = ct_quantity * 1.2; // 계약수량에 1.2 곱하기

            // 입력 필드에 값 설정
            $('#ins_ct_id').val(ct_id);
            $('#ins_item_id').val(ct_item_id);
            $('#ins_lot_size').val(calculated_quantity); // 소수점 없이 설정

            console.log('Set Data:', ct_id, ct_item_id, calculated_quantity.toFixed(2)); // 콘솔에 설정된 데이터 로그

            $(this).prop('checked', false); // 체크박스 해제
        });
        $('input[name="checkAll3"]').prop('checked', false); // 전체 선택 체크박스 해제
    });
});
```

제조지시-등록

제조지시등록

VIEW

담당사원번호	100004	제품번호	10000001	제조LOT번호	300013
계약번호	1000013	공정번호	1005	계획수량	4800
계획착수일	2024-04-02	계획완수일	2024-04-09		

담당사원번호	제품번호	제조LOT번호	계약번호	공정번호	계획수량	계획착수일	계획완수일	삭제
100004	10000001	300013	1000013	1003	4800	2024-04-02	2024-04-09	<button>삭제</button>
100004	10000001	300013	1000013	1004	4800	2024-04-02	2024-04-09	<button>삭제</button>

2024-04-01 ~ 2024-04-05 조회

계약목록

NO	계약번호	제품번호	계약수량	계약체결일	계약출고일
<input type="checkbox"/>	1000013	10000001	4000	2024-04-02	2024-04-09
<input type="checkbox"/>	1000016	10000004	300	2024-04-01	2024-04-07
<input type="checkbox"/>	1000017	10000006	3000	2024-04-01	2024-04-08

취소

Controller

```
@PostMapping("/manufacturingInstructionInsert")
@ResponseBody
public ResponseEntity<?> manufacturingInstructionInsert(
    @RequestBody List<ProcessV0> lots
){
    log.info(lots);
    try {
        processService.insertProInstruction(lots);
        return new ResponseEntity<>(HttpStatus.OK);
    }catch (Exception e) {
        log.error("Error processing instruction insertion", e);
        return new ResponseEntity<>(body: e.getClass().getSimpleName() + " " + e.getMessage(), HttpStatus.INTERNAL_SERVER_ERROR);
    }
}
```

```
function removeRow(button) {
    var row = button.parentNode.parentNode;
    document.getElementById('instructionBody').deleteRow(row.rowIndex - 1);
}
```

localhost:8085 내용:
제조지시 등록에 성공하셨습니다.

확인

제조지시-등록

제조지시 다중 INSERT

JavaScript

```
function addRow() {
    // 입력 필드에서 값 가져오기
    var empId = document.getElementById('ins_emp_id').value;
    var itemId = document.getElementById('ins_item_id').value;
    var lotId = document.getElementById('ins_lot_id').value;
    var ctId = document.getElementById('ins_ct_id').value;
    var piId = document.getElementById('ins_pi_id').value;
    var lotSize = document.getElementById('ins_lot_size').value;
    var startDate = document.getElementById('ins_start_date').value;
    var endDate = document.getElementById('ins_end_date').value;

    // 모든 필드가 채워져 있는지 확인
    if (!empId || !itemId || !lotId || !ctId || !piId || !lotSize || !startDate || !endDate) {
        alert("모든 필드를 채워주세요."); // 필드가 하나라도 비어있으면 경고창 표시
        return; // 함수 실행 종료
    }

    // 테이블 tbody에 새로운 행 추가
    var tableBody = document.getElementById('instructionBody');
    var newRow = tableBody.insertRow();
    newRow.innerHTML = `<td>${empId}</td>
                    <td>${itemId}</td>
                    <td>${lotId}</td>
                    <td>${ctId}</td>
                    <td>${piId}</td>
                    <td>${lotSize}</td>
                    <td>${startDate}</td>
                    <td>${endDate}</td>
                    <td><button onclick="removeRow(this)">삭제</button></td>`;
```

모든 추가된 제조 지시를 서버로 전송하는 함수

JavaScript:ajax

```
function submitInstructions() {
    var tbody = document.getElementById('instructionBody');
    var rows = tbody.rows;
    var instructions = [];

    // 테이블의 모든 행을 순회하면서 각 셀의 내용을 객체에 저장
    for (var i = 0; i < rows.length; i++) {
        var cells = rows[i].cells;
        instructions.push({
            ins_emp_id: parseInt(cells[0].textContent, 10),
            ins_item_id: parseInt(cells[1].textContent, 10),
            ins_lot_id: parseInt(cells[2].textContent, 10),
            ins_ct_id: parseInt(cells[3].textContent, 10),
            ins_pi_id: parseInt(cells[4].textContent, 10),
            ins_lot_size: parseInt(cells[5].textContent, 10),
            ins_start_date: cells[6].textContent,
            ins_end_date: cells[7].textContent
        });
    }

    // AJAX를 사용하여 서버에 데이터 전송
    $.ajax({
        url: '/manufacturingInstructionInsert',
        type: 'POST',
        contentType: 'application/json',
        data: JSON.stringify(instructions),
        success: function(response) {
            alert("제조지시 등록에 성공하셨습니다.");
            document.getElementById('instructionBody').innerHTML = '';
        },
        error: function(xhr, status, error) {
            alert("Failed to register instructions: " + xhr.responseText);
        }
    });
}
```

제조지시-PDF

VIEW

제조지시목록						
	제조LOT번호	계약번호	계획착수일	계획완수일	제품번호	계획수량
<input checked="" type="checkbox"/>	300001	1000013	2024-04-02	2024-04-09	10000001	4800
<input checked="" type="checkbox"/>	300002	1000017	2024-04-01	2024-04-08	10000006	3600
<input checked="" type="checkbox"/>	300003	1000021	2024-04-02	2024-04-09	10000011	4800
<input checked="" type="checkbox"/>	300004	1000024	2024-04-01	2024-04-07	10000016	4800
<input checked="" type="checkbox"/>	300005	1000025	2024-04-02	2024-04-13	10000021	6000

선택된 계약 정보를 서버로부터 조회
체크박스로 선택된 계약 ID들을 서버에 요청하여 해당 계약 정보를 조회하고,
응답 받은 데이터로 테이블을 업데이트

제품 공급 계약 내역서

계약번호	거래처	제품명	계약금	계약수량	계약수량단위	계약체결일	계약출고일
1000013	이케아	의자-A	6000000	4000	ea	2024-04-02	2024-04-09
1000017	코미웨어	침대-A	10000000	3000	ea	2024-04-01	2024-04-08
1000021	레이	소파-A	9700000	4000	ea	2024-04-02	2024-04-09
1000024	에버그린기구	책상-A	5000000	4000	ea	2024-04-01	2024-04-07
1000025	라이트하우스	서랍장-A	4500000	5000	ea	2024-04-02	2024-04-13

계약 당사자 서명: C&R Furniture
날짜: 2024. 4. 21.

PDF로 보기 취소

XML

```
<!--제조지시: 계약 내역 조회 , 제조지시서에서 ajax로 계약 번호 넘기기 위한 쿼리 -->
<select id="selectArrayCt" resultType="com.cnr_furniture.domain.contract.ContractVO" resultMap="mapping_contractVO">
    SELECT c.CT_ID,
        cp.C_NAME,
        i.I_NAME,
        c.CT_AMOUT_MONEY,
        c.CT_QUANTITY,
        c.CT_UNIT,
        to_char(c.CT_DATE, 'yyyy-mm-dd') as CT_DATE,
        to_char(c.CT_OB_DATE, 'yyyy-mm-dd') as CT_OB_DATE
    FROM CONTRACT c
    JOIN COMPANY cp ON c.CT_COMPANY_ID = cp.C_ID <!-- CONTRACT와 COMPANY를 조인: COMPANY ID 기준 -->
    JOIN ITEM i ON c.CT_ITEM_ID = i.I_ID <!-- CONTRACT와 ITEM을 조인: ITEM ID 기준 -->
    WHERE c.CT_DIV = '수주'
        AND c.CT_ID IN
            <!--IN 절의 조건으로 각 아이템 ID를 리스트에서 가져옴 ids 리스트 내 각 아이템을 item 변수로 순회 -->
            <foreach item="item" index="index" collection="ids" open="(" separator="," close=")">
                <!-- 바인딩 변수: 동적으로 아이템 ID를 쿼리에 포함 -->
                #{item}
            </foreach>
    ORDER BY c.CT_ID
</select>
```

Controller

```
// 계약 정보 목록을 저장할 리스트를 초기화
List<ContractVO> contractVOList = new ArrayList<>();
// id 파라미터가 null이 아니고, 공백이 아닌 경우에만 실행
if (id != null && !id.isEmpty()) {
    // id 문자열을 쉼표와 공백 기준으로 분리하여 리스트로 변환
    // ex) 입력된 " 1, 2 , 3" 문자열은 ["1", "2", "3"] 리스트가 됨
    List<String> ids = Arrays.asList(id.split(regex: "\\\s*,\\s*"));

    // 분리된 ID 리스트를 사용하여 데이터베이스에서 해당하는 계약 정보를 조회
    // selectArrayCt: 주어진 ID 리스트에 해당하는 모든 계약 정보를 반환
    contractVOList = processService.selectArrayCt(ids);
}

@GetMapping("/manufacturingInstructionForm")
@ResponseBody
public List<ContractVO> manufacturingInstructionForm(@RequestParam("formattedIds") String formattedIds) {
    // processService의 selectArrayCt 메서드를 호출하여 idList에 해당하는 계약 데이터를 조회
    List<String> idList = Arrays.asList(formattedIds.split(regex: "|"));
    return processService.selectArrayCt(idList);
}
```

IN 절의 조건으로 각 아이템 ID를 리스트에서 가져옴 ids 리스트 내
각 아이템을 item 변수로 순회

AJAX 요청을 통해 호출되며, 주어진 계약 ID 목록에 해당하

는 계약 정보를 조회

동적으로 계약 정보를 로드할 때 사용한다.

@param formattedIds 쉼표로 구분된 계약 ID 목록

JavaScript:ajax

```
function checkboxArrPro() {
    var checkProArr = [];
    if ($('.cityPro:checked').length < 1) {
        alert('계약을 선택하십시오.'); // 계약이 선택되지 않았을 때 경고창
        return;
    }

    $(".cityPro:checked").each(function() {
        var id = $(this).closest('tr').find("#ctProAjax").text().trim(); // 선택된 체크박스에서
        checkProArr.push(id);
    });

    var formattedIds = checkProArr.join(","); // 배열을 쉼표로 구분된 문자열로 변환
    console.log("Selected IDs: " + formattedIds); // 콘솔에 선택된 ID 로그

    $.ajax({
        url: "manufacturingInstructionForm",
        type: "GET",
        data: { formattedIds: formattedIds },
        success: function(response) {
            console.log("Response from server: ", response); // 서버로부터의 응답 로그
            updateTableWithResponse(response); // 테이블 업데이트 함수 호출
        },
        error: function(xhr, status, error) {
            console.error("Error occurred: " + error); // 에러 발생 시 로그
        }
    });
}
```

제조지시-PDF

계약 내역서를 PDF 형태로 생성하고 다운로드

JavaScript

```
function generatePDF() {
  const { jsPDF } = window.jsPDF; // window 객체에서 jsPDF를 추출합니다.
  const doc = new jsPDF(); // 새로운 jsPDF 문서 객체를 생성합니다.

  // 맑은 고딕 폰트 파일을 VFS에 추가하고 폰트를 등록: resources/js/fileFont
  doc.addFileToVFS('malgun.ttf', '_fonts'); // '_fonts'는 폰트의 base64 인코딩된 문자열
  doc.addFont('malgun.ttf', 'malgun', 'normal');
  doc.addFont('malgun.ttf', 'malgun', 'bold'); // 두꺼운 폰트도 등록

  // 문서의 제목 설정 ("계약 내역서")
  doc.setFont('malgun', 'bold'); // 폰트를 두꺼운 맑은 고딕으로 설정
  doc.setFontSize(16);
  const title = "계약 내역서";
  const pageWidth = doc.internal.pageSize.width; // 문서의 페이지 너비 계산
  const titleWidth = doc.getTextWidth(title); // 제목의 텍스트 너비 계산
  doc.text(title, (pageWidth - titleWidth) / 2, 20); // 페이지 중앙 상단에 제목 배치
```

jsPDF라이브러리

```
<script type="text/javascript" src="/resources/js/fileFont/jspdf.js"></script>
<script type="text/javascript" src="/resources/js/fileFont/auto_table.js"></script>
```

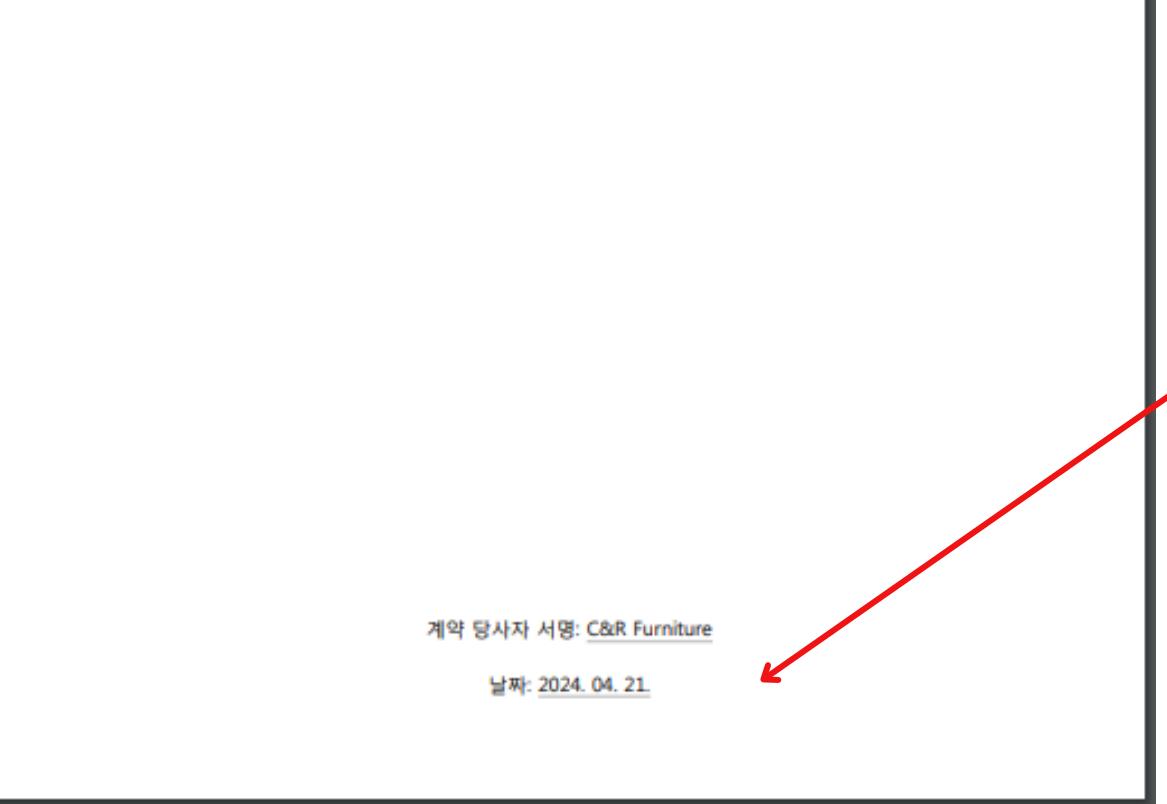
계약 내역서						
계약번호	거래처	제품명	계약금	계약수량	계약수량단위	계약체결일
1000013	이케아	의자-A	60000000	4000	ea	2024-04-02
1000017	코마웨어	침대-A	100000000	3000	ea	2024-04-01
1000021	레이	소파-A	97000000	4000	ea	2024-04-02
1000024	에버그린가구	책상-A	50000000	4000	ea	2024-04-01
1000025	라이트하우스	서랍장-A	45000000	5000	ea	2024-04-02
2024-04-13						

jsPDF 라이브러리 연결



JavaScript autoTable 플러그인을 사용하여 HTML 테이블을 PDF로 변환

```
// autoTable 플러그인을 사용하여 HTML 테이블을 PDF로 변환
if (doc.autoTable) {
  doc.autoTable({
    html: '#resultsTable',
    startY: 30, // 제목 바로 아래에서 테이블 시작
    margin: { bottom: 40 }, // 페이지 하단 여백 설정
    styles: { font: 'malgun', fontStyle: 'normal' }, // 테이블의 폰트 스타일 지정
    didDrawPage: function(data) {
      // 계약 당사자 서명 추가
      // "계약 당사자 서명: " 문자열과 실제 서명("C&R Furniture")
      const signatureText = "계약 당사자 서명: ";
      const signature = "C&R Furniture";
      const fullSignatureText = signatureText + signature; // 전체 서명 문자열 생성
      // 전체 서명 문자열의 너비를 계산
      const signatureTextWidth = doc.getTextWidth(fullSignatureText);
      // 전체 페이지 너비를 기준으로 전체 서명 문자열을 페이지 중앙에 위치시키기 위한 X 좌표를 계산
      const pageWidth = doc.internal.pageSize.width; // 문서의 페이지 너비
      const signatureXPosition = (pageWidth - signatureTextWidth) / 2; // 중앙 정렬을 위한 X 좌표 계산
      // 서명을 페이지의 하단에서 30pt 위에 위치시키기 위한 Y 좌표를 설정
      const signatureYPosition = doc.internal.pageSize.height - 30; // 하단에서 30pt 위
      // 계산된 위치에 전체 서명 문자열을 삽입
      doc.text(fullSignatureText, signatureXPosition, signatureYPosition); // 서명 텍스트 출력
    }
  });
}
```



현재 날짜를 페이지 하단 중앙에 추가

JavaScript

```
// 현재 날짜를 페이지 하단 중앙에 추가
var currentDate = new Date().toLocaleDateString('ko-KR', {
  year: 'numeric', month: '2-digit', day: '2-digit'
}); // 현재 날짜를 '년-월-일' 형식으로 포맷팅
const dateLabel = "날짜: "; // 날짜 라벨
const fullDateText = dateLabel + currentDate; // 전체 날짜 텍스트를 생성
const dateTextWidth = doc.getTextWidth(fullDateText); // 전체 날짜 텍스트의 너비를 계산
const dateXPosition = (pageWidth - dateTextWidth) / 2; // 날짜 텍스트를 페이지 중앙에 위치시키기
const dateYPosition = doc.internal.pageSize.height - 20; // 날짜 텍스트를 페이지 하단에서 20pt 위
doc.text(fullDateText, dateXPosition, dateYPosition); // 날짜 텍스트를 지정된 위치에 출력

// 날짜에 밀줄 추가 (굵게)
const dateWidth = doc.getTextWidth(currentDate); // 실제 날짜 ('YYYY-MM-DD')의 텍스트 너비를 계산
// 날짜 라벨 너비를 추가한 위치에서 시작하여 날짜 너비만큼 오른쪽으로 이동한 위치에 밀줄을 추가
doc.line(dateXPosition + doc.getTextWidth(dateLabel), dateYPosition + 1,
  dateXPosition + doc.getTextWidth(dateLabel) + dateWidth, dateYPosition + 1);

// 생성된 PDF 파일을 'contract-details.pdf'라는 이름으로 저장
doc.save('contract-details.pdf');

} else {
  console.error('autoTable function is not available.');
}
}
```

제조수행지시

제조수행지시

조회 등록

제조LOT번호: 지시일자: 연도-월-일 ~ 연도-월-일

공정번호:

NO	제조LOT번호	공정번호	제품번호	계획수량	생산시작날짜	진행사항	비고
1	300001	1004	10000001	4800	2024-04-22	생산대기	테스트
2	300007	1013	10000031	6000	2024-04-23	생산대기	test
3	300007	1014	10000031	6000	2024-04-23	생산대기	test
4	300001	1001	10000001	4800	2024-04-02	생산완료	
5	300001	1002	10000001	4800	2024-04-02	생산완료	
6	300001	1003	10000001	4800	2024-04-02	생산완료	
7	300001	1004	10000001	4800	2024-04-02	생산완료	
8	300001	1005	10000001	4800	2024-04-02	생산완료	
9	300001	1006	10000001	4800	2024-04-02	생산완료	
10	300001	1007	10000001	4800	2024-04-02	생산완료	

제조수행지시등록

제조LOT번호	--선택--	제품번호	--선택--
공정번호	--선택--	계획수량	
비고			

추가

제조LOT번호	제품번호	공정번호	계획수량	비고	삭제

등록

취소

제조수행지시

VIEW

Controller

```
// 현재 선택된 제품번호와 공정 번호에 기반한 LOT 번호 목록을 조회
List<ProcessVO> insLotList = processService.selectLotIdsByItemAndProcessId();
List<ProcessVO> insItemList = null; // 선택된 LOT ID에 따른 제품 목록 초기화
List<ProcessVO> insPiList = null; // 선택된 제품 번호에 따른 공정 번호 목록 초기화
// LOT 번호를 선택했을 경우
if (processDate.getIns_lot_id() > 0) {
    insItemList = processService.selectItemsByLotId(processDate.getIns_lot_id());
    // 선택된 LOT 번호에서 제품 아이디를 선택했을 경우
    if (processDate.getIns_item_id() > 0) {
        insPiList = processService.selectProcessIdsByItemAndLotId(processDate.getIns_lot_id(), processDate.getIns_item_id());
    } else {
        insPiList = List.of(); // 제품 번호가 선택되지 않았을 경우 빈 목록 할당
    }
} else {
    insItemList = List.of(); // LOT 번호가 선택되지 않았을 경우 빈 목록 할당
    insPiList = List.of(); // 동일하게 빈 목록 할당
}
```

제조수행지시등록

제조LOT번호	300001	제품번호	10000001
공정번호	1005	계획수량	
비고			

등록 취소

```
// LOT 번호가 선택되어 있으면
if (proLot) {
    $.ajax({
        url: '/insItemList?ins_lot_id=' + proLot, // 서버에 제품 번호 요청
        type: 'GET',
        dataType: 'json',
        success: function(data) {
            console.log('response data', data);
            if (data.length > 0) {
                proItemSelect.append($('', { value: "", text: "--선택--"}));
                $.each(data, function(index, item) {
                    proItemSelect.append($('', {
                        value: item.ins_item_id,
                        text: item.ins_item_id // 제품 번호 옵션 추가
                    }));
                });
            } else {
                proItemSelect.append('<option value="">No Options Available</option>'); // 옵션이 없을 경우
            }
        },
        error: function(xhr, status, error) {
            console.error("Error loading proItem options: ", error);
        }
    });
} else {
    proItemSelect.append('<option value="">Select Lot first</option>'); // LOT 먼저 선택하라는 메시지
    proPiSelect.append('<option value="">Select Item first</option>'); // ITEM 먼저 선택하라는 메시지
}
```

JavaScript: ajax

```
// 제품 번호가 변경되었을 때의 이벤트 핸들러
$('#proItem').on('change', function() {
    var proLot = $('#proLot').val(); // LOT 번호를 다시 가져옴
    var proItem = $(this).val(); // 선택된 ITEM 번호를 가져옴
    var proPiSelect = $('#proPi'); // 공정 번호 select element
    proPiSelect.empty(); // 공정 번호 선택지 초기화

    // 제품 번호가 선택되어 있으면
    if (proItem) {
        $.ajax({
            url: '/insPiList?ins_lot_id=' + proLot + '&ins_item_id=' + proItem, // 서버에 공정 번호 요청
            type: 'GET',
            dataType: 'json',
            success: function(data) {
                console.log('response data', data);
                if (data.length > 0) {
                    proPiSelect.append($('', { value: "", text: "--선택--"}));
                    $.each(data, function(index, pi) {
                        proPiSelect.append($('', {
                            value: pi.ins_pi_id,
                            text: pi.ins_pi_id
                        }));
                    });
                } else {
                    proPiSelect.append('<option value="">No Options Available</option>'); // 옵션이 없을 경우
                }
            },
            error: function(xhr, status, error) {
                console.error("Error loading proPi options: ", error);
            }
        });
    } else {
        proPiSelect.append('<option value="">Select Item first</option>'); // ITEM 먼저 선택하라는 메시지
    }
});
```

JavaScript: ajax

300007

--선택--

--선택--

1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024

JavaScript

제조수행지시등록

VIEW

제조LOT번호	300001	제품번호	10000001
공정번호	1001	계획수량	4800
비고	TEST		

추가

제조LOT번호	제품번호	공정번호	계획수량	비고	삭제
300001	10000001	1002	4800	TEST	삭제
300002	10000006	1013	4800	TEST	삭제

등록 취소

```

@PostMapping("/manufacturingPerformInsert")
@ResponseBody
public ResponseEntity<?> manufacturingPerformInsert(
    @RequestBody List<ProcessRunVO> runs
) {
    log.info(runs);
    try {
        processService.insertProcessDa(runs);
        return new ResponseEntity<>(HttpStatus.OK);
    } catch (Exception e) {
        log.error("Error processing instruction insertion", e);
        return new ResponseEntity<>(body: e.getClass().getSimpleName() + " " + e.getMessage(), HttpStatus.INTERNAL_SERVER_ERROR);
    }
}

function submitManufacturingInstructions() {
    var instructions = [];
    // 테이블의 모든 행을 순회하면서 데이터 객체 생성
    $('#manufacturingDataEntryTable tbody tr').each(function() {
        var row = $(this);
        instructions.push({
            p_lot_id: parseInt(row.find('td:eq(0)').text()),
            p_b_item_id: parseInt(row.find('td:eq(1)').text()),
            p_pi_id: parseInt(row.find('td:eq(2)').text()),
            p_plan_quantity: parseInt(row.find('td:eq(3)').text()),
            p_note: row.find('td:eq(4)').text()
        });
    });
    console.log("Submitting Instructions:", JSON.stringify(instructions));

    // AJAX를 통해 서버에 제조 지시 데이터 전송
    $.ajax({
        url: '/manufacturingPerformInsert',
        type: 'POST',
        contentType: 'application/json',
        data: JSON.stringify(instructions),
        success: function(response) {
            Swal.fire('게시가 완료되었습니다.', '', 'success');
            $('#manufacturingDataEntryTable tbody').empty();
        },
        error: function(xhr, status, error) {
            Swal.fire('Error', 'Failed to register instructions: ' + xhr.responseText, 'error');
        }
    });
}

```

공정정보관리 - 공정정보등록

공정정보

공정번호:

설비번호:

설비명:

설비유형:

설비목록

NO	설비번호	설비명	설비유형
1	6000052	오비탈 샌더4호	샌딩기
2	6000064	오비탈 샌더 5호	샌딩기
3	6000001	디테일 샌더4호	샌딩기
4	6000005	디테일 샌더 5호	샌딩기
5	6000025	벨트 샌더4호	샌딩기
6	6000038		

설비등록:

설비등록

설비번호(설비명) 입력:

설비목록:

설비번호	설비명	설비유형
6000076	오비탈 샌더4호	샌딩기
6000077	오비탈 샌더 5호	샌딩기
6000078	디테일 샌더4호	샌딩기
6000079	디테일 샌더 5호	샌딩기
6000080	벨트 샌더4호	샌딩기

추가된 설비목록:

설비번호	설비명	설비유형

등록 버튼:

공정정보관리 - 공정정보등록

Controller

```

no usages
@GetMapping("/searchManagementVO")
@ResponseBody
public List<ManagementVO> searchManagementVO(@RequestParam(value = "miId", required = false) Integer miId) {
    List<ManagementVO> managementVOList = processService.selectByMiId(miId);
    return managementVOList;
}
no usages
@GetMapping("/processInfoAjax")
@ResponseBody
public List<ManagementVO> processInfoAjax() {
    return processService.selectM();
}

```

공정정보등록

The screenshot shows the 'Management Information Registration' page. At the top, there are tabs for '공정번호' (Process Number), '공정번호' (Process Number), '설비번호' (Equipment Number), and '설비번호' (Equipment Number). The '설비번호' input field contains '6000071'. Below the tabs is a search bar with '설비번호(설비명) 입력' (Enter Equipment Number/Equipment Name), a '조회' (Search) button, and a '리셋' (Reset) button. A red arrow points from the '설비번호' input field to a table titled '설비목록' (Equipment List) on the left. Another red arrow points from the '설비번호' input field to a table titled '추가된 설비목록' (Added Equipment List) on the right. The '설비목록' table contains rows for equipment numbers 6000072 through 6000076. The '추가된 설비목록' table contains rows for equipment numbers 6000069, 6000070, and 6000071. At the bottom are buttons for '공정등록' (Process Registration), '설비수정' (Equipment Modification), and '취소' (Cancel).

	설비번호	설비명	설비유형
<input type="checkbox"/>	6000072	자동 대폐기5호	대폐기
<input type="checkbox"/>	6000073	자동 대폐기6호	대폐기
<input type="checkbox"/>	6000074	유압 클드 프레스 4호	집성기
<input type="checkbox"/>	6000075	유압 클드 프레스 5호	집성기
<input type="checkbox"/>	6000076	오비탈 샌더4호	샌더기

	설비번호	설비명	설비유형
<input checked="" type="checkbox"/>	6000069	엣지 밴더5호	밴딩기
<input type="checkbox"/>	6000070	복합 면취기3호	면취기
<input type="checkbox"/>	6000071	복합 면취기4호	면취기

VIEW

The screenshot shows the 'Management Information Registration' page with a list of added services. The list is enclosed in a red box. It contains 20 items, each with a checkbox and a service ID. The first few items are: 6000066, 6000066 재단기5호; 6000067, 6000067 재단기6호; 6000068, 6000068 엣지 밴더4호; 6000069, 6000069 엣지 밴더5호; 6000070, 6000070 복합 면취기3호; 6000071, 6000071 복합 면취기4호; 6000072, 6000072 자동 대폐기5호; 6000073, 6000073 자동 대폐기6호; 6000074, 6000074 유압 클드 프레스 4호; 6000075, 6000075 유압 클드 프레스 5호; 6000076, 6000076 오비탈 샌더4호; 6000077, 6000077 오비탈 샌더 5호; 6000078, 6000078 오비탈 샌더 6호.

	설비번호	설비명	설비유형
<input checked="" type="checkbox"/>	6000066	재단기5호	
<input checked="" type="checkbox"/>	6000067	재단기6호	
<input checked="" type="checkbox"/>	6000068	엣지 밴더4호	
<input checked="" type="checkbox"/>	6000069	엣지 밴더5호	
<input checked="" type="checkbox"/>	6000070	복합 면취기3호	
<input checked="" type="checkbox"/>	6000071	복합 면취기4호	
<input checked="" type="checkbox"/>	6000072	자동 대폐기5호	
<input checked="" type="checkbox"/>	6000073	자동 대폐기6호	
<input checked="" type="checkbox"/>	6000074	유압 클드 프레스 4호	
<input checked="" type="checkbox"/>	6000075	유압 클드 프레스 5호	
<input checked="" type="checkbox"/>	6000076	오비탈 샌더4호	
<input checked="" type="checkbox"/>	6000077	오비탈 샌더 5호	
<input checked="" type="checkbox"/>	6000078	오비탈 샌더 6호	

```

// 추가된 설비 목록에서 추가 버튼을 눌렀을 때 설비 목록으로 데이터를 입력해주는 이벤트
$('#addSelected2').click(function() {
    console.log('Button clicked');

    $('#addedMng input:checkbox').each(function() {
        var row = $(this).closest('tr');
        var machineId = $.trim(row.find('td.miId').text()); // miId 클래스를 가진 td에서 텍스트 추출 후 공백 제거

        console.log('Extracted Machine ID:', machineId); // 콘솔에 추출된 machineId 출력

        $('#pi_machine_id').val(machineId); // 설비번호 입력 필드에 값 설정
        if ($('#pi_machine_id').val() === machineId) {
            console.log('Value set successfully');
        } else {
            console.log('Failed to set value:', $('#pi_machine_id').val());
        }
    });

    row.find('input[type="checkbox"]').prop('checked', false);
});

$('#checkAll1').prop('checked', false);
$('#checkAll2').prop('checked', false);
});

// 리셋 검색
function resetSearch() {
    $.ajax({
        url: '/processInfoAjax',
        type: 'GET',
        dataType: 'json',
        success: function(data) {
            $('#miId').val('');
            updateTable(data, '#mng');
            $('#addedMng').empty(); // 추가된 설비목록 비우기
        },
        error: function(xhr, status, error) {
            console.error('Reset AJAX Error:', status, error, xhr);
        }
    });
}

// 테이블 업데이트
function updateTable(data, selector) {
    var tableContent = '';
    $.each(data, function(i, managementVO) {
        tableContent += '<tr>' +
            '<td><input type="checkbox" name="check1"></td>' +
            '<td class="miId">' + managementVO.miId + '</td>' +
            '<td>' + managementVO.miName + '</td>' +
            '<td>' + managementVO.miType + '</td>' +
            '</tr>';
    });
    $(selector).html(tableContent);
}

```

JavaScript: ajax

느낀 점(조다혜)

[좋았던 점]

- 개발 과정에서 팀원들이 작성한 코드를 서로 검토하면서 다양한 프로그래밍 스타일과 해결 방안을 보고 배울 수 있어 좋았음.
- 해결하기 어려운 문제가 생기면 팀원들과 함께 논의 하여 해결 방안을 모색함으로써 효율적으로 시간을 관리 할 수 있 었음.
- 설계에 많은시간을 투자한만큼 개발에 대한 각자의 이해도가 높았던 점이 좋았음.

[아쉬웠던 점]

- 개발기간이 짧아 설계를 탄탄하게 한 대신 개발 기간이 적어 다양한 기능을 구사하지 못했던 것이 아쉬움 이 남았음.

배지현

생산관리, 작업관리, 작업자관리, 당일작업조회

생산관리

- 생산현황, 공정별 설비가동현황
- 공정별 설비가동현황의 경우 현재 진행되고 있는 제조LOT번호가 업데이트되어 조회될 수 있도록 함.

작업관리

- 공정별 제조수행정보, 작업목록, 생산실적, 자재투입내역, 작업자조회
- 자재투입 등록/수정
- 작업등록, 작업상세 및 수정/삭제

작업자관리

- 작업자배치 목록 조회
- 작업자관리 - 작업자 CRUD

당일작업조회

- 사원이 유일하게 접근할 수 있는 작업관리 페이지
- 공정별 설비상태, 당일작업목록 조회

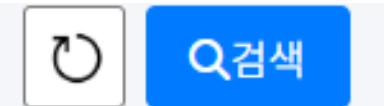
검색

작업관리

거래처번호	<input type="text"/>
제조 LOT번호	<input type="text"/>

- 200001 나무마켓 (발주업체)
- 200002 원목킹 (발주업체)
- 200003 소나무가게 (발주업체)
- 200004 원목스토어 (발주업체)
- 200005 루트나무 (발주업체)

조회일자
연도-월-일 ~ 연도-월-일
작업번호



대부분의 검색항목을
datalist를 이용해 해당
번호의 정보를 확인하고
가져올 수 있도록 한다.

검색을 위한 메소드 생성

모든 페이지가 검색을 필요로 하기
때문에 메소드를 따로 만들어 사용
하여 불필요한 코드를 줄인다.

```
/**  
 * Desc: 검색을 위한 메소드  
 */  
3 usages ▲ jihyeon *  
private void workSearch(WorkSearchVO workSearchVO, Model model) {  
    model.addAttribute(attributeName: "companyList", workService.findCompanyList());  
    model.addAttribute(attributeName: "itemList", workService.findItemList());  
    model.addAttribute(attributeName: "instructionList", workService.findInstructionList());  
    model.addAttribute(attributeName: "processInfoList", workService.findProcessInfoList());  
    model.addAttribute(attributeName: "machineInfoList", workService.findMachineInfoList());  
    model.addAttribute(attributeName: "workList", workService.findWorkList());  
    model.addAttribute(attributeName: "dpNameList", workService.findDpNameList());  
    model.addAttribute(attributeName: "empInfoList", workService.findEmpInfoList());  
}
```

```
@GetMapping("/work")  
public String workSearch(WorkSearchVO workSearchVO,  
                        Model model){  
  
    workSearch(workSearchVO, model); // 검색을 위한 메소드 사용
```

검색에 필요한 VO 생성

```
/**  
 * Desc: Work의 검색을 위한 VO  
 */  
29 usages ▲ jihyeon  
@Getter  
@Setter  
public class WorkSearchVO {  
    private String type; // 이름/번호 선택  
    private String keyword; // 검색 입력값  
    private String find_work_company; // 거래처 검색  
    private String find_work_item; // 제품 검색  
    private String workStartDate; // 시작 날짜  
    private String workEndDate; // 종료 날짜  
    private String find_work_instruction; // 제조 지시의 제조LOT번호  
    private String find_work_processInfo; // 공정번호  
    private String find_work_id; // 작업번호  
    private String find_dp_name; // 작업팀  
    private String find_work_machine; // 설비번호  
    private String find_emp_id; // 사원번호  
  
    private List<WorkSelectCompanyVO> companyList; // 회사 리스트  
    private List<WorkSelectItemVO> itemList; // 제품 리스트  
    private List<WorkSelectInstructionVO> instructionList; // 제조지시 정보 리스트  
    private List<WorkSelectProcessInfoVO> processInfoList; // 공정 정보 리스트  
    private List<WorkSelectMachineInfoVO> machineInfoList; // 설비 정보 리스트  
    private List<WorkSelectWorkVO> workList; // 작업 리스트  
    private List<WorkVO> dpNameList; // 부서명 리스트  
    private List<WorkSelectEmpInfoVO> empInfoList; // 사원정보 리스트  
  
    no usages ▲ jihyeon  
    public String[] getTypeArr() { return type == null? new String[] {} : type.split(regex: ""); }
```

공정관리 - 작업관리 (1) - 조회 및 작업등록 (1)

1. work의 작업목록 위 작업등록 버튼 클릭시

작업목록														
NO	작업번호	제조LOT번호	지시일자	작업팀	공정번호	공정명	설비명	작업위치	제품번호	제품명	진행상황	규격	단위	소요시간(분)
1	8	300003	2024-04-23	생산 3팀	1025	원자재준비	지게차4호	3-1-1	10000011	소파-A	작업전	1인용	EA	540
2	7	300001	2024-04-04	생산 1팀, 생산 2팀	1001	원자재준비	지게차1호	1-1-1	10000001	의자-A	작업중	성인용	EA	480
3	6	300001	2024-04-18	생산 1팀, 생산 3팀	1001	원자재준비	지게차1호	1-1-1	10000001	의자-A	작업종료	성인용	EA	480
4	5	300001	2024-04-17	생산 1팀	1011	적재	지게차2호	1-9-1	10000001	의자-A	작업종료	성인용	EA	600
5	4	300001	2024-04-17	생산 1팀	1011	적재	지게차2호	1-9-1	10000001	의자-A	작업종료	성인용	EA	480

2. 작업등록 모달창이 나타난다.

제조로트번호 입력

→ 제품번호, 제품명, 생산단위 자동채우기

공정번호 입력

→ 설비번호, 설비명, 작업위치, 남은생산수량 자동채우기

작업등록

제조LOT번호	<input type="text"/>	공정번호	<input type="text"/> 공정번호	공정명	<input type="text"/>
설비번호	<input type="text"/>	설비명	<input type="text"/>	작업위치	<input type="text"/>
제품번호	<input type="text"/>	제품명	<input type="text"/>	생산단위	<input type="text"/>
남은생산수량	<input type="text"/>	현재작업계획수량	<input type="text"/>	소요시간(분)	<input type="text"/>
작업시작시간	<input type="text"/> 연도-월-일 -- ::--	작업종료시간	<input type="text"/> 연도-월-일 -- ::--		
비고	<input type="text"/>				

등록 취소

[JavaScript 코드]

로트번호 입력에 따른 공정번호 리스트 조회와 자동채우기를 위한 Js 코드

```
// [로트번호] 입력에 따른 [공정번호] 조회
$.ajax({
    url: 'workInsertModalProIdByLotId', // 이 URL은 Backend에서 처리할 경로
    type: 'GET',
    data: {workInsertModalLotId: workInsertModalLotId},
    success: function(data) { // 성공 시
        console.log('workInsertModalLotId : ' + workInsertModalLotId);
        $('#workInsertModalProcessId').empty();
        let str = '<option value="">공정번호</option>';
        $.each(data, function(i){
            str += '<option value="' + data[i].workInsertModalProcessId + '">' +
                + data[i].workInsertModalProcessId + '</option>';
        });
        $('#workInsertModalProcessId').append(str);
    },
    error: function(xhr, status, error) { // 실패 시
        console.log("Error workInsertModal lotId proId : ", error);
    }
}); // /.[로트번호] 입력에 따른 [공정번호] 조회
```

```
// [로트번호] 입력에 따른 [제품번호],[제품명],[생산단위] 조회 및 자동채우기
$.ajax({
    url: 'workInsertModalAutoDataByLotId', // 이 URL은 Backend에서 처리할 경로
    type: 'GET',
    data: {workInsertModalLotId: workInsertModalLotId},
    success: function(data) { // 성공 시
        $('#workInsertModalItemId').val(data.workInsertModalItemId);
        $('#workInsertModalItemName').val(data.workInsertModalItemName);
        $('#workInsertModalUnit').val(data.workInsertModalUnit);
    },
    error: function(xhr, status, error) { // 실패 시
        console.log("Error workInsert lotId AutoData : ", error);
    }
}); // /.[로트번호] 입력에 따른 [제품번호],[제품명],[생산단위] 자동채우기
```

공정관리 - 작업관리 (2) - 조회 및 작업등록 (2)

3. 소요시간 계산 : 작업종료시간 - 작업시작시간

조건 1. 작업시간 차이가 810분 (13시간 30분) 이상인 경우 -90분

조건 2. 작업시간 차이가 4시간 이상 8시간 미만인 경우 -30분

조건 3. 작업시간 차이가 4시간 미만인 경우 -0분

그 외. -60분

[JavaScript 코드]

소요시간 계산 함수

```
// 소요시간 계산 : 작업종료시간 - 작업시작시간
var startTime = $('#workInsertModalWorkStartTime').val();
var endTime = $('#workInsertModalWorkEndTime').val();

// 시작시간과 종료시간이 둘 다 입력된 경우에만 계산 수행
if (startTime && endTime) {
    // ISO8601 포맷의 날짜 문자열을 Date 객체로 변환
    var startDate = new Date(startTime);
    var endDate = new Date(endTime);

    // 두 시간의 차이를 밀리초 단위로 계산 후 분 단위로 변환
    var durationMinutes = (endDate - startDate) / 60000;
    var id='';

    // 작업시간 차이가 810분 (13시간 30분) 이상인 경우
    if (durationMinutes >= 810) {
        durationMinutes -= 90;
    } else if (durationMinutes >= 240 && durationMinutes < 480) {
        // 작업시간 차이가 4시간 이상 8시간 미만인 경우
        durationMinutes -= 30;
    } else if (durationMinutes < 240) {
        // 작업시간 차이가 4시간 미만인 경우 - 추가적인 시간 감소 없이 순수한 차이만 적용
        // 이 경우에는 durationMinutes가 변경되지 않으므로, 아무 조치를 취하지 않음
    } else {
        // 그 외의 경우 (작업시간 차이가 13시간 30분 미만이고 4시간 미만이 아닌 경우)
        durationMinutes -= 60;
    }

    // 결과를 소요시간 입력 필드에 설정
    $('#workInsertModalWorkTime').val(durationMinutes.toFixed(0));
}
```

4. 해당 모달창의 추가 버튼 클릭 시 입력한 데이터가 배열에 저장된다.

+추가

작업등록목록

NO	제조LOT번호	공정번호	공정명	설비번호	설비명	작업위치	제품번호	제품명	생산단위	남은생산수량	현재작업계획수량
1	300002	1012	원자재준비	6000054	지게차3호	2-1-1	10000006	침대-A	EA	3600	1200

[WorkController 코드]

ajax를 통해 받은 데이터로 작업등록 처리

```
/*
 * Desc: Work 의 작업 등록 시, DB 저장 - [작업 테이블]
 * @param arrays 자재투입목록 클라이언트로부터 JSON 형태로 받아 WorkInsertModalVO 객체 리스트로 변환
 */
no usages ± jihyeon *
@PostMapping(value = "/M/process/workInsertArr")
@ResponseBody
public ResponseEntity<?> workInsert(
    @RequestBody List<WorkInsertModalVO> arrays
) {
    log.info(arrays);
    try {
        workService.workInsert(arrays);
        return ResponseEntity.ok().body(Map.of( k1: "success", v1: true,
                                                k2: "message", v2: "등록이 성공되었습니다."));
    } catch (Exception e) {
        log.error("등록 실패, Error: {}" + e.getMessage() + e);
        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).body(Map.of(
            k1: "success", v1: false,
            k2: "message", v2: "등록에 실패하였습니다. 에러: " + e.getMessage()));
    }
}
```

공정관리 - 작업관리 (3) - 작업 상세 및 수정, 삭제 (1)

1. work의 작업목록 중 작업상세 버튼 클릭시

작업목록																	
NO	작업번호	제조LOT번호	지시일자	작업됨	공정번호	공정명	설비명	작업위치	제품번호	제품명	진행상황	규격	단위	소요시간(분)	계획수량	생산수량	작업상세
1	20	300001	2024-04-24		1002	오버레이	유압 콜드 프레스 3호	1-2-1	10000001	의자-A	작업중	성인용	EA	540	1200	1300	작업상세

2. 작업상세 조회와 수정, 삭제를 할 수 있는 모달창이 나타난다.

앞서 작업등록에 이용된 것처럼 작업종료시간 - 작업시작시간으로 소요시간을 계산한다.

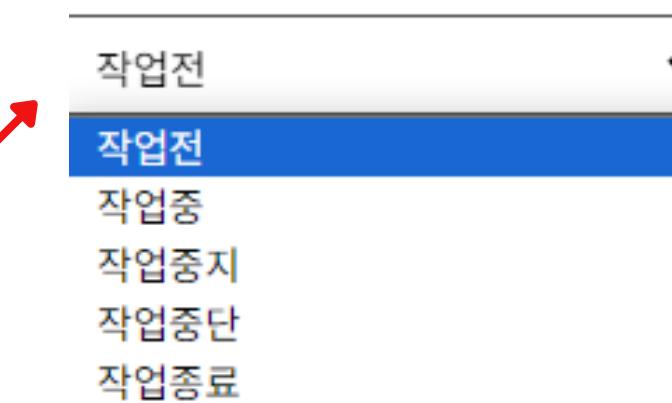
작업시작시간이 현재시간보다 과거라면 이미 작업이 시작된것으로 readonly 속성을 가진다.

작업상세 및 수정

작업번호	20	제조LOT번호	300001			
공정번호	1002	공정명	오버레이			
설비번호	6000064	설비명	유압 콜드 프레스	작업위치	1-2-1	
제품번호	10000001	제품명	의자-A	생산단위	EA	
현재작업계획수량	1200	생산수량	1300	규격	성인용	
작업시작시간	2024-04-24 오후 06:00	작업종료시간	2024-04-25 오전 04:00			
소요시간(분)	540	진행상황	작업중			
비고						

수정할 경우 한번에 수정되어야 할 사항

- 생산수량이 10 이상일 경우 process 테이블의 누적생산수량에 합해진다.
- 진행상황이 '작업중' 일 경우 설비 작동 테이블의 MW_STATUS = '가동'
- 진행상황이 '작업정지' 혹은 '작업종료'일 경우 설비 작동 테이블의 MW_STATUS = '비가동'
- 진행상황이 '작업중단'일 경우 설비 작동 테이블의 MW_STATUS = '수리중', MW_CONDITION = '수리요청'



삭제 수정 취소

공정관리 - 작업관리 (3) - 작업 상세 및 수정, 삭제 (2)

[WorkMapper.interface 코드]

수정에 필요한 메소드 생성

```
/**  
 * Desc: Work 의 작업상세 - 모달창을 이용한 작업 수정  
 */  
2 usages ▲ jihyeon  
void workDetailModalUpdate(WorkDetailModalVO workDetailModalVO);  
  
/**  
 * Desc: Work 의 작업상세 - 모달창을 이용한 제조수행 수정  
 * work 테이블의 W_PLAN_QUANTITY가 수정됐을 때 이전값을 빼주고 현재값을 더해준다.  
 * -> P_ITEM_QUANTITY = P_ITEM_QUANTITY + #{생산수량}  
 */  
2 usages ▲ jihyeon  
void workDetailModalUpdateProQuantity(WorkDetailModalVO workDetailModalVO);  
  
/**  
 * Desc: Work 의 작업상세 - 모달창을 이용한 설비 작동상태 수정 (work의 w_status가 작업중일 때  
 * -> MW_STATUS = '가동'  
 */  
2 usages ▲ jihyeon  
void workDetailModalUpdateMiWorkByStatusWork(WorkDetailModalVO workDetailModalVO);  
  
/**  
 * Desc: Work 의 작업상세 - 모달창을 이용한 설비 작동상태 수정 (work의 w_status가 작업중단일 때  
 * -> MW_STATUS = '수리/중' , MW_CONDITION = '수리요청'  
 */  
2 usages ▲ jihyeon  
void workDetailModalUpdateMiWorkByStatusRepair(WorkDetailModalVO workDetailModalVO);  
  
/**  
 * Desc: Work 의 작업상세 - 모달창을 이용한 설비 작동상태 수정 (work의 w_status가 작업정지 또는 작업종료일 때  
 * -> MW_STATUS = '비/가동'  
 */  
2 usages ▲ jihyeon  
void workDetailModalUpdateMiWorkByStatusRestFin(WorkDetailModalVO workDetailModalVO);
```

[WorkMapper.xml 코드]

MyBatis를 이용하여 데이터베이스 쿼리 <-> 프로그래밍 언어 코드를 분리

```
<!-- Work 의 작업상세 - 모달창을 이용한 작업 수정 -->  
<update id="workDetailModalUpdate" parameterType="com.cnr_furniture.domain.work.workMNG.WorkDetailModalVO">  
    UPDATE  
        work  
    SET  
        W_START_TIME = #{workDetailModalWorkStartTime},  
        W_END_TIME = #{workDetailModalWorkEndTime},  
        W_TIME = #{workDetailModalWorkTime},  
        W_STATUS = #{workDetailModalWorkStatus},  
        W_PLAN_QUANTITY = #{workDetailModalPlanQuantity},  
        W_ITEM_QUANTITY = #{workDetailModalProQuantity},  
        W_NOTE = #{workDetailModalNote, jdbcType=VARCHAR}  
    WHERE  
        w_id = #{workDetailModalWorkId}  
        and w_lot_id = #{workDetailModalLotId}  
        and w_pi_id = #{workDetailModalProcessId}  
</update>  
  
<!-- Work 의 작업상세 - 모달창을 이용한 제조수행 수정 : work 테이블의 W_PLAN_QUANTITY가 수정됐을 때 이전값을 빼주고 현재값을 더함 -->  
<update id="workDetailModalUpdateProQuantity" parameterType="com.cnr_furniture.domain.work.workMNG.WorkDetailModalVO">  
    UPDATE PROCESS  
    SET  
        P_ITEM_QUANTITY = P_ITEM_QUANTITY + #{workDetailModalProQuantity}  
    WHERE  
        P_LOT_ID = #{workDetailModalLotId}  
        and P_PI_ID = #{workDetailModalProcessId}  
</update>  
  
<!-- Work 의 작업상세 - 모달창을 이용한 제조수행 수정 : work 테이블의 W_PLAN_QUANTITY가 수정됐을 때 이전값을 빼주고 현재값을 더함 -->  
<update id="workDetailModalUpdateProQuantity" parameterType="com.cnr_furniture.domain.work.workMNG.WorkDetailModalVO">  
    UPDATE PROCESS  
    SET  
        P_ITEM_QUANTITY = P_ITEM_QUANTITY + #{workDetailModalProQuantity}  
    WHERE  
        P_LOT_ID = #{workDetailModalLotId}  
        and P_PI_ID = #{workDetailModalProcessId}  
</update>  
  
<!-- Work 의 작업상세 - 모달창을 이용한 설비 작동상태 수정 : work 테이블의 w_status가 작업중일 때 MW_STATUS = '가동' -->  
<update id="workDetailModalUpdateMiWorkByStatusWork" parameterType="int">  
    UPDATE MACHINE_WORK  
    SET  
        MW_STATUS = '가동'  
    WHERE  
        MW_MI_ID = #{workDetailModalMachineId}  
</update>  
  
<!-- Work 의 작업상세 - 모달창을 이용한 설비 작동상태 수정 : work 테이블의 w_status가 작업중단일 때 MW_STATUS = '수리중', MW_CONDITION = '수리요청' -->  
<update id="workDetailModalUpdateMiWorkByStatusRepair" parameterType="int">  
    UPDATE MACHINE_WORK  
    SET  
        MW_STATUS = '수리중',  
        MW_CONDITION = '수리요청'  
    WHERE  
        MW_MI_ID = #{workDetailModalMachineId}  
</update>
```

공정관리 - 작업관리 (3) - 작업 상세 및 수정, 삭제 (3)

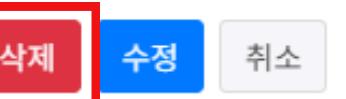
[WorkService.interface 코드]

MyBatis를 이용해 작성한 쿼리를 트랜잭션 처리하는 메소드

```
/**  
 * Desc: Work 의 작업상세 - 모달창을 이용한 작업 수정 및 설비 작동상태, 제조수행 테이블 수정  
 */  
1 usage 1 implementation ━ jihyeon  
void workDetailModalUpdate(WorkDetailModalVO workDetailModalVO);
```

제품번호	10000001	제품명	의자-A	생산단위	EA
현재작업계획수량	1200	생산수량	1300	규격	성인용
작업시작시간	2024-04-24 오후 06:00	작업종료시간	2024-04-25 오전 04:00		
소요시간(분)	540	진행상황	작업중		
비고					

Ajax를 통해 삭제 처리



```
// 작업 삭제 이벤트 핸들러  
$('#workDetailDeleteBtn').click(function() {  
    if (confirm('이 작업을 삭제하시겠습니까?')) {  
        $.ajax({  
            url: '/M/process/workDetailDelete', // 서버의 삭제 처리 경로  
            type: 'POST',  
            data: {  
                workDetailModalWorkId: workDetailModalWorkId  
            },  
            success: function(response) {  
                if (response.success) {  
                    alert('작업이 성공적으로 삭제되었습니다.');// 성공 후, '/work' 페이지로 리다이렉트  
                } else {  
                    alert('작업 삭제에 실패했습니다: ' + response.message);  
                }.bind(this), // 현재 클릭한 버튼에 대한 참조를 유지  
                error: function(xhr, status, error) {  
                    alert('삭제 과정에서 오류가 발생했습니다: ' + error);  
                }  
            }  
        }); // workDetailDelete ajax
    }
});
```

[WorkServiceImpl 구현체 코드]

업데이트 할 조건을 한번에 트랜잭션 처리

```
@Override  
@Transactional  
트랜잭션 어노테이션 이용  
public void workDetailModalUpdate(WorkDetailModalVO workDetailModalVO) {  
    try {  
        // 기존 생산 수량 정보 가져오기 (예를 들어 데이터베이스 쿼리)  
        int oldProQuantity = fetchOldProductionQuantity(workDetailModalVO.getWorkDetailModalWorkId());  
        // 작업 테이블 업데이트  
        workMapper.workDetailModalUpdate(workDetailModalVO);  
        생산수량 변경사항 검사  
        if (workDetailModalVO.getWorkDetailModalProQuantity() != oldProQuantity) {  
            workDetailModalVO.setWorkDetailModalProQuantity(  
                workDetailModalVO.getWorkDetailModalProQuantity() - oldProQuantity  
            );  
            workMapper.workDetailModalUpdateProQuantity(workDetailModalVO);  
        }  
        작업 상태에 따른 설비 상태 업데이트  
        // 작업 상태에 따른 설비 상태 업데이트  
        switch (workDetailModalVO.getWorkDetailModalWorkStatus()) {  
            case "작업중":  
                workMapper.workDetailModalUpdateMiWorkByStatusWork(workDetailModalVO);  
                break;  
            case "작업중단":  
                workMapper.workDetailModalUpdateMiWorkByStatusRepair(workDetailModalVO);  
                break;  
            case "작업정지":  
            case "작업종료":  
                workMapper.workDetailModalUpdateMiWorkByStatusRestFin(workDetailModalVO);  
                break;  
            default:  
                // 예외 처리 또는 기타 상태 처리  
                log.error("알 수 없는 작업 상태입니다: " + workDetailModalVO.getWorkDetailModalWorkStatus());  
        }  
    } catch (Exception e) {  
        log.error(message: "작업 상세 업데이트 중 에러 발생", e);  
        // 트랜잭션 롤백이 자동으로 발생합니다.  
        throw e;  
    }  
}
```

공정관리 - 작업관리 (4) - 자재투입 등록 (1)

1. work의 작업목록 위 자재투입 버튼 클릭시

⊕ 자재투입

2. 자재투입등록 모달창이 나타난다.

자재투입등록

제조LOT번호	<input type="text"/>	공정번호	공정번호	<input type="button"/>	
제품번호	<input type="text"/>	제품명	<input type="text"/>	계획생산수량	<input type="text"/>
자재번호	자재번호	자재명	<input type="text"/>	제품1EA별 투입수량	<input type="text"/>
최소투입수량	<input type="text"/>	단위	<input type="text"/>	투입수량	<input type="text"/>

추가

등록 취소

3. 제조LOT번호 입력 → 해당 제조LOT번호에 따른 공정번호 조회

→ 제품번호, 제품명, 계획생산수량 데이터 자동채우기

→ 불러온 제품번호에 따른 BOM의 자재번호 목록 조회

제조LOT번호	300001	공정번호	공정번호	<input type="button"/>													
제품번호	10000001	제품명	의자-A	계획생산수량	4800												
자재번호	자재번호	자재번호	<input type="button"/>	<table border="1"><tr><td>공정번호</td></tr><tr><td>1001</td></tr><tr><td>1002</td></tr><tr><td>1003</td></tr><tr><td>1004</td></tr><tr><td>1005</td></tr><tr><td>1006</td></tr><tr><td>1007</td></tr><tr><td>1008</td></tr><tr><td>1009</td></tr><tr><td>1010</td></tr><tr><td>1011</td></tr></table>		공정번호	1001	1002	1003	1004	1005	1006	1007	1008	1009	1010	1011
공정번호																	
1001																	
1002																	
1003																	
1004																	
1005																	
1006																	
1007																	
1008																	
1009																	
1010																	
1011																	
최소투입수량	20000001 20000002 20000003 20000004 20000005																

[JavaScript 코드와 workController]

제조LOT번호 입력에 따른 공정번호 목록 조회

```
// [로트번호] 입력에 따른 [공정번호] 조회
$.ajax({
  url: '/insertMaterialModalInsProId', // 이 URL은 Backend에서 처리할 경로
  type: 'GET',
  data: {insLotIdModal: insLotId},
  success: function(data) { // 성공 시
    $('#processIdModal').empty();
    let str = '<option value="">공정번호</option>';
    $.each(data, function(i){
      str += '<option value="' + data[i].processIdModal + '">' + data[i].processIdModal + '</option>';
    });
    $('#processIdModal').append(str);
  },
  error: function(xhr, status, error) { // 실패 시
    console.log("Error insertMaterial lotId proId : ", error);
  }
}); // [. [로트번호] 입력에 따른 [공정번호] 조회
```

```
@GetMapping("/insertMaterialModalInsProId")
@ResponseBody
public List<WorkInsertMaterialModalVO> getInsProIdModal(
  @RequestParam("insLotIdModal") int insLotIdModal
) {
  return workService.getInsProIdModal(insLotIdModal);
}
```

제조LOT번호 입력에 따른 제품번호, 제품명, 계획생산수량 데이터 자동채우기

```
// [로트번호] 입력에 따른 [제품번호],[제품명],[계획생산수량] 조회 및 자동채우기
$.ajax({
  url: '/insertMaterialModalInsProIdAutoData', // 이 URL은 Backend에서 처리할 경로
  type: 'GET',
  data: {insLotIdModal: insLotId},
  success: function(data) { // 성공 시
    $('#itemIdModal').val(data.itemIdModal);
    $('#itemNameModal').val(data.itemNameModal);
    $('#productionPlanQuantityModal').val(data.productionPlanQuantityModal);
  },
  error: function(xhr, status, error) { // 실패 시
    console.log("Error insertMaterial lotId AutoData : ", error);
  }
}); // [. [로트번호] 입력에 따른 [제품번호],[제품명],[계획생산수량] 자동채우기
```

```
@GetMapping("/insertMaterialModalInsProIdAutoData")
@ResponseBody
public WorkInsertMaterialModalVO getInsLotIdAutoDataModal(
  @RequestParam("insLotIdModal") int insLotIdModal
) {
  return workService.getInsLotIdAutoDataModal(insLotIdModal);
}
```

공정관리 - 작업관리 (4) - 자재투입 등록 (2)

4. 자재번호 선택

→ 해당 자재번호에 따른

자재명, 제품1EA별 투입수량, 최소투입수량, 단위 데이터 자동채우기

자재번호	20000001	자재명	소나무	제품1EA별 투입수량	1
최소투입수량	4800	단위	M	투입수량	

5. 투입수량 입력 → 추가버튼 클릭

→ 추가할 자재투입내역을 미리 보여주는 테이블 생성

자재투입등록

제조LOT번호	300001	공정번호	1001				
제품번호	10000001	제품명	의자-A	계획생산수량	4800		
자재번호	20000001	자재명	소나무	제품1EA별 투입수량	1		
최소투입수량	4800	단위	M	투입수량	4800		
<button>⊕ 추가</button>							

☰ 자재투입목록

NO	투입일자	제조LOT번호	공정번호	제품번호	제품명	계획생산수량	자재번호	자재명	제품1EA별 투입수량	최소투입수량	단위	투입수량	삭제
1	2024-04-21	300001	1001	10000001	의자-A	4800	20000001	소나무	1	4800	M	4800	삭제

등록 취소

5. 자재투입목록의 삭제 버튼 클릭 시 배열에 저장된 자재투입내역이 삭제된다.

[JavaScript 코드]

추가 버튼 클릭 시 데이터를 수집하여 자재투입목록에 미리 보여주는 함수

```
/* 모달창에서 입력된 데이터 수집하는 함수 */
function collectDataFromModal() {
    // 모달창에서 입력된 데이터 수집
    return {
        invDate: new Date().toISOString().split('T')[0],
        insLotId: $('#insLotIdModal').val(),
        processId: $('#processIdModal').val(),
        itemId: $('#itemIdModal').val(),
        itemName: $('#itemNameModal').val(),
        productionPlanQuantity: $('#productionPlanQuantityModal').val(),
        materialId: $('#materialIdModal').val(),
        materialName: $('#materialNameModal').val(),
        item1EaMaterial: $('#item1EaMaterialModal').val(),
        minInsertQuantity: $('#minInsertQuantityModal').val(),
        unitMaterial: $('#unitMaterialModal').val(),
        insertQuantity: $('#insertQuantityModal').val()
    };
} // .collectDataFromModal
```

```
/* 목록(Table)에 데이터(행) 추가하는 함수 */
function addToInsertMaterialList(data) {
    // 테이블
    var table = $('.newWorkMaterialList tbody');

    // 테이블 행 추가
    // [주의]: data.컬럼명의 '컬럼명'은 collectDataFromModal() 함수의 컬럼명과 일치시킬 것.
    var newRow = `<tr>
        <td class="listSeq">${table.children().length + 1}</td> // 추가된 목록 번호
        <td class="invDate">${data.invDate}</td> // 투입날짜
        <td class="insLotId">${data.insLotId}</td> // LOT번호
        <td class="processId">${data.processId}</td> // 공정번호
        <td class="itemId">${data.itemId}</td> // 제품번호
        <td class="itemName">${data.itemName}</td> // 제품명
        <td class="productionPlanQuantity">${data.productionPlanQuantity}</td> // 계획생산수량
        <td class="materialId">${data.materialId}</td> // 자재번호
        <td class="materialName">${data.materialName}</td> // 자재명
        <td class="item1EaMaterial">${data.item1EaMaterial}</td> // 제품1EA별 투입수량
        <td class="minInsertQuantity">${data.minInsertQuantity}</td> // 최소투입수량
        <td class="unitMaterial">${data.unitMaterial}</td> // 단위
        <td class="insertQuantity">${data.insertQuantity}</td> // 투입수량
        <td onclick="removeRow(this);> // [삭제] 버튼
            style="cursor:pointer;
            color:#c82333;
            text-align:center;">
                삭제
            </td>
        </tr>`;
    table.append(newRow);
}
```

공정관리 - 작업관리 (4) - 자재투입 등록 (3)

7. 추가된 자재투입목록 확인 후 등록 버튼 클릭 시 등록된다.

☰ 자재투입목록

NO	투입일자	제조LOT번호	공정번호	제품번호	제품명	계획생산수량	자재번호	자재명	제품1EA별 투입수량	최소투입수량	단위	투입수량	삭제
1	2024-04-21	300001	1001	10000001	의자-A	4800	20000001	소나무	1	4800	M	4800	삭제

등록 취소

[JavaScript 코드]

등록 버튼 클릭시 데이터를 수집해 배열에 저장 후 서버로 전송하여 등록한다.

```
/* 등록 - 테이블에서 데이터 수집 */
function collectArrFromTable() {
    var arrays = [];
    $('.newWorkMaterialList tbody tr').each(function() {
        var row = $(this);
        arrays.push({
            invDateModal: row.find('.invDate').text(),
            insLotIdModal: row.find('.insLotId').text(),
            processIdModal: row.find('.processId').text(),
            itemIdModal: row.find('.itemId').text(),
            itemNameModal: row.find('.itemName').text(),
            productionPlanQuantityModal: row.find('.productionPlanQuantity').text(),
            materialIdModal: row.find('.materialId').text(),
            materialNameModal: row.find('.materialName').text(),
            item1EaMaterialModal: row.find('.item1EaMaterial').text(),
            minInsertQuantityModal: row.find('.minInsertQuantity').text(),
            unitMaterialModal: row.find('.unitMaterial').text(),
            insertQuantityModal: row.find('.insertQuantity').text(),
        });
    });
    return arrays;
}
```

[WorkController 코드]

```
@PostMapping(value = "/insertMaterialArr")
@ResponseBody
public ResponseEntity<?> workInsertMaterial(
    @RequestBody List<WorkInsertMaterialModalVO> arrays
) {
    log.info(arrays);
    try {
        workService.workInsertMaterial(arrays);
        return ResponseEntity.ok().body(Map.of( k1: "success", v1: true, k2: "message", v2: "등록이 성공되었습니다."));
    } catch (Exception e) {
        log.error("등록 실패, Error: {}" + e.getMessage() + e);
        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).body(Map.of( k1: "success", v1: false,
            k2: "message", v2: "등록에 실패하였습니다. 에러: " + e.getMessage()));
    }
}
```

```
/* 등록 - 데이터베이스 저장 로직 */
function insertMaterialArr(arrays) {
    $.ajax({
        url: '/insertMaterialArr', // 데이터 저장을 처리하는 서버의 API 경로
        type: 'POST',
        contentType: 'application/json',
        data: JSON.stringify(arrays),
        success: function(response) {
            alert('등록이 완료되었습니다.');
            window.location.href = '/work'; // 저장 후, '/work'로 리다이렉트
        },
        error: function(xhr, status, error) {
            console.error('자재투입등록 Error', error);
        }
    });
}
```

공정관리 - 작업관리 (5) - 자재투입 수정 (1)

1. work의 자재투입내역 중 수정 버튼 클릭시

☰ 자재투입내역											수정
1	300001	1001	10000001	의자-A	이케아	2024-04-02	20000001	소나무	M	4800	수정

2. 자재투입수정 모달창이 나타난다.

자재투입수정

제조LOT번호	300001	공정번호	1001		
제품번호	10000001	제품명	의자-A	단위	M
자재번호	20000001	자재명	소나무	투입수량	4800

3. 투입수량을 수정하여 수정 버튼 클릭 시 수정된다. ←

[JavaScript 코드]

수정버튼을 통해 데이터를 가져와 자재투입수정 모달창에 불러온다.

```
/* 수정 - Ajax로 해당 행의 내용 조회 */
$(document).on('click', '.insertMatEditBtn', function() {
    var inv_lot_id = $(this).data('lot-id'); // 수정 버튼에서 inv_lot_id 값 가져오기
    var inv_pi_id = $(this).data('pi-id'); // 수정 버튼에서 inv_pi_id 값 가져오기
    var inv_material_id = $(this).data('mat-id'); // 수정 버튼에서 inv_material_id 값 가져오기
    var inv_quantity = $(this).data('inv-quantity'); // 수정 버튼에서 inv_quantity 값 가져오기
    console.log('선택된 inv_lot_id: ', inv_lot_id);
    console.log('선택된 inv_pi_id: ', inv_pi_id);
    console.log('선택된 inv_material_id: ', inv_material_id);
    console.log('선택된 inv_quantity: ', inv_quantity);

    $.ajax({
        url: '/insertMaterialForUpdateModal?inv_lot_id=' + inv_lot_id + '&inv_pi_id=' + inv_pi_id + '&inv_material_id=' + inv_material_id + '&inv_quantity=' + inv_quantity,
        type: 'GET',
        data: {
            inv_lot_id: inv_lot_id,
            inv_pi_id: inv_pi_id,
            inv_material_id: inv_material_id,
            inv_quantity: inv_quantity
        },
        success: function(data) {
            $('#inv_lot_id').val(data.inv_lot_id);
            $('#inv_pi_id').val(data.inv_pi_id);
            $('#ins_item_id').val(data.ins_item_id);
            $('#i_name').val(data.i_name);
            $('#b_unit').val(data.b_unit);
            $('#inv_material_id').val(data.inv_material_id);
            $('#m_name').val(data.m_name);
            $('#inv_quantity').val(data.inv_quantity);
            $('#updateInvQuantity').val(data.inv_quantity);

            $('#workMaterialInsertEditModal').modal('show'); // 모달창 보여주기
        },
        error: function() {
            console.log("수정 모달창의 데이터 조회 실패 Error", error);
            alert('데이터를 불러오는데 실패했습니다.');
        }
    });
});
```

공정관리 - 작업관리 (5) - 자재투입 수정 (1)

[JavaScript 코드]

투입수량을 수정 후 수정 버튼 클릭 시 불러온 데이터를 이용하여 수정한다.

```
/* 수정 - Ajax로 DB에 업데이트 */
$('#insertMatModalUpdateBtn').on('click', function() {
    if(!confirm('수정하시겠습니까?')) {
        return; // 사용자가 '아니오'를 선택하면 할수를 종료
    }

    // Form에서 데이터 수집
    var formData = {
        inv_lot_id: $('#inv_lot_id').val(),
        inv_pi_id: $('#inv_pi_id').val(),
        inv_material_id: $('#inv_material_id').val(),
        inv_quantity: $('#inv_quantity').val(),
        updateInvQuantity: $('#updateInvQuantity').val()
    };
    console.log(formData);

    // Ajax 요청을 사용하여 서버에 수정 사항 전송
    $.ajax({
        url: '/insertMaterialUpdate', // 수정 처리를 위한 서버 URL
        type: 'POST', // HTTP 요청 방식
        contentType: 'application/json', // 요청 컨텐츠 타입
        data: JSON.stringify(formData), // JSON 형식으로 데이터 전송
        success: function(response) {
            if(response.success) {
                alert('수정이 완료되었습니다.');
                window.location.href = '/work'; // 성공 후, 'work' 페이지로 리다이렉트
            } else {
                alert('수정 실패: ' + response.message);
            }
        },
        error: function(xhr, status, error) { // 요청 실패 시
            alert('수정 중 에러가 발생했습니다. 에러: ' + error);
        }
    }); // ./insertMaterialUpdate 의 ajax
});
```

[SQL문을 이용해 조회할 메소드]

```
/**
 * 조건에 따른 수정할 자재투입내역 조회
 * Desc: Work 의 자재투입내역 데이터를 가져오는 메소드
 */
2 usages ± jihyeon
WorkSelectInsertMaterialVO workSelectInsertMaterialForUpdate(Map<String, Object> map);
```

[수정 모달창에 띄울 데이터를 조회하는 SQL문]

```
SELECT
    ROWNUM AS rn, mi.inv_LOT_ID, mi.inv_PI_ID, mi.ins_item_id, mi.i_name,
    mi.ct_company_id, mi.c_name, mi.inv_date, mi.INV_MATERIAL_ID,
    mi.m_name, mi.b_unit, mi.inv_quantity
FROM
    (SELECT
        DISTINCT inv.inv_LOT_ID, inv.inv_PI_ID, ins.INS_ITEM_ID, it.i_name,
        ct.CT_Company_ID, c.c_name,
        to_char(inv.INV_DATE,'yyyy-mm-dd') AS inv_date,
        inv.INV_MATERIAL_ID, m.M_NAME, upper(b.B_UNIT) AS B_UNIT,
        substr(inv.INV_QUANTITY,2) AS INV_QUANTITY
    FROM
        INVENTORY inv
        JOIN INSTRUCTION ins ON inv.inv_LOT_ID = ins.INS_LOT_ID
        JOIN PROCESS_INFO pi ON ins.ins_PI_ID = pi.PI_ID
        JOIN CONTRACT ct ON ins.INS_CT_ID = ct.CT_ID
        JOIN COMPANY c ON ct.CT_Company_ID = c.C_ID
        JOIN ITEM it ON ins.INS_ITEM_ID = it.I_ID
        JOIN BOM b ON inv.INV_MATERIAL_ID = b.B_MATERIAL_ID
        JOIN MATERIAL m ON inv.INV_MATERIAL_ID = m.M_ID
    WHERE inv.INV_LOT_ID IS NOT NULL
        AND inv.INV_PI_ID IS NOT NULL
        AND inv.INV_QUANTITY < 0
    ORDER BY
        inv.INV_LOT_ID, inv.inv_PI_ID, inv.INV_MATERIAL_ID) mi
WHERE
    mi.INV_LOT_ID = #{inv_lot_id}
    AND mi.INV_PI_ID = #{inv_pi_id}
    AND mi.INV_MATERIAL_ID = #{inv_material_id}
    AND mi.INV_QUANTITY = #{inv_quantity}
```

공정관리 - 작업관리 (5) - 자재투입 수정 (2)

[WorkService 인터페이스]

```
/**  
 * Desc: Work 의 조건에 따른 수정할 자재투입내역 데이터를 가져오는 메소드  
 */  
  
1 usage 1 implementation ✎ jihyeon  
WorkSelectInsertMaterialVO getInsertMaterialForUpdate(int inv_lot_id,  
                                                     int inv_pi_id,  
                                                     int inv_material_id,  
                                                     int inv_quantity);
```

[WorkServiceImpl 구현체]

VO로 만들지 않고 불러오는 값을 Map에 담아 받아온다.

```
/** Desc: Work 의 조건에 따른 수정할 자재투입내역 데이터를 가져오는 메소드 */  
  
1 usage ✎ jihyeon *  
@Override  
public WorkSelectInsertMaterialVO getInsertMaterialForUpdate(int inv_lot_id,  
                                                               int inv_pi_id,  
                                                               int inv_material_id,  
                                                               int inv_quantity) {  
  
    Map<String, Object> paramInsertMaterialForUpdate = new HashMap<>();  
    paramInsertMaterialForUpdate.put("inv_lot_id", inv_lot_id);  
    paramInsertMaterialForUpdate.put("inv_pi_id", inv_pi_id);  
    paramInsertMaterialForUpdate.put("inv_material_id", inv_material_id);  
    paramInsertMaterialForUpdate.put("inv_quantity", inv_quantity);  
    return workMapper.workSelectInsertMaterialForUpdate(paramInsertMaterialForUpdate);  
}
```

[WorkController]

조회한 값이 수정 모달창으로 넘어오는지 디버깅을 통해 확인하며, 넘어올 경우 수정된다.

```
/**  
 * 자재투입수정  
 * Desc: Work 의 자재투입내역 - 모달창을 이용한 자재투입 수정  
 * @param: workUpdateMaterialModalVO - 업데이트 할 조건과 데이터를 담은 VO 객체  
 */  
  
no usages ✎ jihyeon  
@PostMapping("/insertMaterialUpdate")  
public ResponseEntity<?> workInsertMaterialUpdate(  
    @RequestBody WorkUpdateMaterialModalVO workUpdateMaterialModalVO  
) {  
    try {  
        log.info("수정될 내용: " + workUpdateMaterialModalVO);  
        log.info("수정할 inv_lot_id: " + workUpdateMaterialModalVO.getInv_lot_id());  
        log.info("수정할 inv_pi_id: " + workUpdateMaterialModalVO.getInv_pi_id());  
        log.info("수정할 inv_material_id: " + workUpdateMaterialModalVO.getInv_material_id());  
        log.info("수정할 inv_quantity: " + workUpdateMaterialModalVO.getInv_quantity());  
        log.info("수정된 updateInvQuantity: " + workUpdateMaterialModalVO.getUpdateInvQuantity());  
  
        workService.updateWorkInsertMaterial(workUpdateMaterialModalVO);  
        return ResponseEntity.ok().body(Map.of( k1: "success", v1: true, k2: "message", v2: "업데이트가 성공적으로 완료되었습니다."));  
    } catch(Exception e) {  
        log.error( message: "수정 실패: ", e);  
        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR)  
            .body(Map.of( k1: "success", v1: false, k2: "message", v2: "업데이트 실패: " + e.getMessage()));  
    }  
}
```

공정관리 - 작업자관리 - 등록 / 상세 및 수정, 삭제 (1)

1. workerInsert 의 작업자배치 중 관리 버튼 클릭 시

작업자배치											
NO	작업번호	작업시작시간	작업종료시간	제조LOT번호	공정번호	공정명	설비번호	설비명	작업위치	작업자	관리
1	7	2024-04-05 09:00:00	2024-04-05 18:00:00	300001	1001	원자재준비	6000052	지게차1호	1-1-1		관리
2	6	2024-04-04 09:00:00	2024-04-04 18:00:00	300001	1001	원자재준비	6000052	지게차1호	1-1-1	100009(이지은), 100042(박서진)...	관리
3	5	2024-04-17 19:00:00	2024-04-18 07:00:00	300001	1011	작재	6000053	지게차2호	1-9-1	100014(이도현), 100106(정유빈), 100110(박시은), 1...	관리

2. 작업자관리 모달창이 나타난다.

→ 조회된 데이터를 통해 작업자 등록을 할 수 있다.

→ 부서명 선택 시 부서명에 따라 사원번호 datalist 조회

작업자관리

작업번호	6				
제조LOT번호	300001	공정번호	1001	설비번호	6000052
작업시작시간	2024-04-04 오전 09:00		작업종료시간	2024-04-04 오후 06:00	
부서명	생산 1팀		사원번호	<input style="width: 100%; height: 30px; border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;" type="text"/> + 추가	

100013
생산 1팀 홍수진(디렉터)

100009
생산 1팀 이지은(매니저)

100014
생산 1팀 이도현(매니저)

100032
생산 1팀 최태민(사원)

100042
생산 1팀 박서진(사원)

100046
생산 1팀 정하은(사원)

100054
생산 1팀 이도현(사원)

100055
생산 1팀 임서연(사원)

100064
생산 1팀 이재원(사원)

100072
생산 1팀 최유진(사원)

100075
생산 1팀 박민준(사원)

100084
생산 1팀 김준서(사원)

100088
생산 1팀 최하윤(사원)

100090
생산 1팀 이서준(사원)

100106
생산 1팀 정유빈(사원)

등록된 작업자 목록				
부서명	사원번호	사원명	직급	삭제
생산 1팀	100009	이지은	매니저	삭제
생산 1팀	100042	박서진	사원	삭제

추가된 작업자 목록		
사원번호	삭제	

등록
취소

[JavaScript 코드]

작업자배치 중 작업자관리 모달창으로 가져올 데이터 조회

→ 불러온 작업번호에 따른 제조LOT번호, 공정번호, 설비번호, 작업시작시간, 작업종료시간 조회 후 데이터 자동채우기

```
/*
 * 작업자 관리 모달창 - Ajax로 해당 행의 내용 조회 */
$('.workerInsertBtn').click(function() {
    var edit_w_id = $(this).data('work-id'); // 등록 버튼에서 w_id 값 가져오기
    var edit_w_lot_id = $(this).data('lot-id'); // 등록 버튼에서 w_lot_id 값 가져오기
    var edit_w_pi_id = $(this).data('pi-id'); // 등록 버튼에서 w_pi_id 값 가져오기
    var edit_pi_machine_id = $(this).data('mi-id'); // 등록 버튼에서 pi_machine_id 값 가져오기
    var edit_w_start_time = $(this).data('start-time'); // 등록 버튼에서 w_start_time 값 가져오기
    var edit_w_end_time = $(this).data('end-time'); // 등록 버튼에서 w_end_time 값 가져오기
    var allempList = $(this).data('emp-list'); // 등록 버튼에서 allempList 값 가져오기

    var allempLength = allempList.length; // 작업자 목록 조회 시 데이터 존재여부 판별을 위함

    console.log('선택된 edit_w_id: ', edit_w_id);
    console.log('선택된 edit_w_lot_id: ', edit_w_lot_id);
    console.log('선택된 edit_w_pi_id: ', edit_w_pi_id);
    console.log('선택된 edit_pi_machine_id: ', edit_pi_machine_id);
    console.log('선택된 edit_w_start_time: ', edit_w_start_time);
    console.log('선택된 edit_w_end_time: ', edit_w_end_time);

    /* [작업번호]에 따른 [제조LOT번호], [공정번호], [설비번호], [작업시작시간], [작업종료시간] 조회 */
    $.ajax({
        url: '/workerInsertModalSelectData?w_id=' + edit_w_id,
        type: 'GET',
        data: {
            edit_w_id: edit_w_id,
            edit_w_lot_id: edit_w_lot_id,
            edit_w_pi_id: edit_w_pi_id,
            edit_pi_machine_id: edit_pi_machine_id,
            edit_w_start_time: edit_w_start_time,
            edit_w_end_time: edit_w_end_time
        },
        success: function(data) {
            $('#edit_w_id').val(data.edit_w_id);
            $('#edit_w_lot_id').val(data.edit_w_lot_id);
            $('#edit_w_pi_id').val(data.edit_w_pi_id);
            $('#edit_pi_machine_id').val(data.edit_pi_machine_id);
            $('#edit_w_start_time').val(data.edit_w_start_time);
            $('#edit_w_end_time').val(data.edit_w_end_time);

            $('#workerInsertModal').modal('show'); // 모달창 보여주기
        },
        error: function() {
            console.log("작업자관리 모달창의 데이터 조회 실패 Error", error);
            alert('데이터를 불러오는데 실패했습니다.');
        }
    }); // ./workerInsertModalSelectData 의 ajax
})
```

공정관리 - 작업자관리 - 등록 / 상세 및 수정 (2) - 상세 및 수정(1)

[JavaScript 코드]

부서명 선택에 따른 사원 정보 조회

```
/* 부서명 선택에 따른 사원번호 조회 */
$("#select[name=edit_dp_name]").change(function() {
  var edit_dp_name = $(this).val();
  console.log('edit_dp_name : ' + edit_dp_name);
  $.ajax({
    url: '/workerInsertModalEmpInfoByDpName', // 이 URL은 Backend에서 처리할 경로
    type: 'GET',
    data: {edit_dp_name: edit_dp_name},
    success: function(data) { // 성공 시
      console.log('edit_dp_name : ' + data.edit_dp_name);
      $('#workerInsertEmpInfoList').empty();
      let str = '<datalist id="workerInsertEmpInfoList">';
      $.each(data, function(i){
        str += '<option value="' + data[i].edit_emp_id + '">' + data[i].edit_dp_name + ' ' +
               + data[i].edit_e_name + '(' + data[i].edit_e_role +')'+ '</option>';
      });
      str += '</datalist>';
      $('#edit_emp_id').after(str);
    },
    error: function(xhr, status, error) { // 실패 시
      console.log("Error workerInsert DpName EmpId : ", error);
    }
  }); // ./ 부서명 선택에 따른 사원번호 조회 ajax
}); // ./ select[name=edit_dp_name] change function
```

[JavaScript 코드]

작업번호에 따른 작업자 정보 조회

```
/* 작업번호에 따른 작업자 정보 조회 */
$.ajax({
  url: '/workerInsertModalWorkerInfoByWorkId', // 이 URL은 Backend에서 처리할 경로
  type: 'GET',
  data: {edit_w_id: edit_w_id},
  success: function(data) { // 성공 시
    console.log('edit_w_id : ' + data.edit_w_id);
    // 테이블
    var table = $('.workerInsertModalWorkerInfoByWorkIdList tbody');
    table.empty();
    var tbody = '';
    if(allemplLength!=0){
      $.each(data, function(i){
        tbody += '<tr class="'+data[i].workerInfoEmpId+'>' +
                '<td>' + data[i].workerInfoDpName + '</td>'; // 부서명
        tbody += '<td>' + data[i].workerInfoEmpId + '</td>'; // 사원번호
        tbody += '<td>' + data[i].workerInfoEmpName + '</td>'; // 사원명
        tbody += '<td>' + data[i].workerInfoEmpRole + '</td>'; // 직급
        tbody += '<td class="delete-button" data-emp-id="' + data[i].workerInfoEmpId +
                '" data-work-id="' + edit_w_id +
                '" style="cursor:pointer; color:#c82333; text-align:center;">삭제</td>';
        tbody += '</tr>';
      });
    } else {
      tbody += '<tr><td colspan="5">조회된 내용이 없습니다.</td></tr>';
    }
    table.append(tbody);
  },
  error: function(xhr, status, error) { // 실패 시
    console.log("Error workerInsert WorkId WorkerInfo : ", error);
  }
}); // ./ 작업번호에 따른 작업자 정보 조회 ajax
```

3. 작업번호에 따른 작업자 정보 조회

→ 등록된 작업자 목록에 조회한 데이터가 나타난다.

등록된 작업자 목록

부서명	사원번호	사원명	직급	삭제
생산 1팀	100009	이지은	매니저	삭제
생산 1팀	100042	박서진	사원	삭제

공정관리 - 작업자관리 - 등록 / 상세 및 수정, 삭제 (2) - 상세 및 수정(2)

4. 사원번호를 입력 후 추가 버튼 클릭 시

작업자관리

작업번호	6				
제조LOT번호	300001	공정번호	1001	설비번호	6000052
작업시작시간	2024-04-04 오전 09:00	작업종료시간	2024-04-04 오후 06:00		
부서명	생산 1팀	사원번호	100054		

⊕ 추가

5. 추가된 작업자 목록에 입력한 사원번호가 등록된다.

등록된 작업자 목록				
부서명	사원번호	사원명	직급	삭제
생산 1팀	100009	이지은	매니저	삭제
생산 1팀	100042	박서진	사원	삭제

추가된 작업자 목록	
사원번호	삭제
100046	삭제
100054	삭제

등록 **취소**

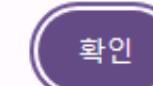
6. 추가된 작업자 목록은 등록 버튼 클릭 시 등록된다.

등록된 작업자 목록				
부서명	사원번호	사원명	직급	삭제
생산 1팀	100042	박서진	사원	삭제
생산 1팀	100046	정하은	사원	삭제
생산 1팀	100054	이도현	사원	삭제

단, 등록한 작업자를 추가하려고 할 경우
아래와 같이 alert창을 통해 추가를 방지한다.

localhost:8085 내용:

이미 목록에 존재하는 데이터입니다.



[JavaScript 코드]

추가 버튼 클릭 시 중복 데이터를 검사하여 등록 전 추가 방지하는 함수

```
/* 작업자관리 모달창 - [추가] 버튼 클릭 이벤트 */
$('#addInsertWorkerBtn').click(function() {
    console.log('작업자관리 추가 버튼 클릭'); // 디버깅을 위한 코드
    // 모달창에서 입력된 데이터 수집
    var data = collectDataFromModal();
    console.log('작업자관리 모달창에서 입력된 데이터: ', data);

    // 입력 필드 유효성 검사
    if(!validateFields(data)) {
        return; // 유효성 검사 실패 시 추가 작업 중단
    }

    // 중복 데이터 검사
    if (isDuplicateData(data)) {
        alert('이미 목록에 존재하는 데이터입니다.');
        $('#edit_emp_id').val('');
        return; // 중복 데이터 존재 시 추가 작업 중단
    }

    console.log('유효성 검사 데이터: ', data); // 유효성 검사 data 디버깅

    // 데이터 목록에 추가
    addToInserWorkerList(data);
    clearInputFields();

}); // /.작업자관리 모달창 - [추가] 버튼 클릭 이벤트

/* 중복 데이터 검사 */
function isDuplicateData(data) {
    var isDuplicate = false;
    var existingEmpIds = [];

    // 이미 등록된 작업자 목록 검사
    $('.workerInsertModalWorkerInfoByWorkIdList tbody tr').each(function() {
        var empId = $(this).find('td:nth-child(2)').text(); // 사원번호 칼럼
        existingEmpIds.push(empId);
    });

    // 추가된 작업자 목록 검사
    $('.workerInsertModalWorkerInsertList tbody tr').each(function() {
        var empId = $(this).find('td:nth-child(1)').text(); // 사원번호 칼럼
        existingEmpIds.push(empId);
    });

    // 입력된 데이터의 사원번호가 기존 목록에 있는지 확인
    if (existingEmpIds.indexOf(data.empIdModal) !== -1) {
        isDuplicate = true;
    }

    return isDuplicate;
}
```

공정관리 - 작업자관리 - 등록 / 상세 및 수정, 삭제 (3) - 등록

[JavaScript 코드]

모달창에서 입력된 데이터를 수집하는 함수

→ input type이 datetime-local일 경우 value 값이 '2024-04-21 T09:00'으로 받아오기 때문에 T를 제외한 값을 수집한다.

```
/* 모달창에서 입력된 데이터 수집하는 함수 */
function collectDataFromModal() {
    // 모달창에서 입력된 데이터 수집
    return {
        wIdModal: $('#edit_w_id').val(), // 작업번호
        wStartTimeModal: $('#edit_w_start_time').val().substr(0,10)+ // 작업시작시간
                        +'$('#edit_w_start_time').val().substr(11), // 작업시작시간
        wEndTimeModal: $('#edit_w_end_time').val().substr(0,10)+ // 작업종료시간
                        +'$('#edit_w_end_time').val().substr(11), // 작업종료시간
        empIdModal: $('#edit_emp_id').val() // 사원번호
    };
}
```

추가된 작업자 목록의 데이터를 수집하는 함수

```
/* 등록 - 테이블에서 데이터 수집 */
function collectArrFromTable() {
    var arrays = [];
    $('.workerInsertModalWorkerInsertList tbody tr').each(function() {
        var row = $(this);
        console.log('wIdModal : '+row.find('.wIdModal').val());
        console.log('wStartTimeModal : '+row.find('.wStartTimeModal').val());
        console.log('wEndTimeModal : '+row.find('.wEndTimeModal').val());
        console.log('empIdModal : '+row.find('.empIdModal').text());
        arrays.push({
            edit_w_id: row.find('.wIdModal').val(),
            edit_w_start_time: row.find('.wStartTimeModal').val(),
            edit_w_end_time: row.find('.wEndTimeModal').val(),
            edit_w_emp_id: row.find('.empIdModal').text()
        });
    });
    return arrays;
}
```

추가한 작업자를 추가된 작업자 목록에 추가하는 함수

```
/* 목록(Table)에 데이터(행) 추가하는 함수 */
function addToInserWorkerList(data) {
    // 테이블
    var table = $('.workerInsertModalWorkerInsertList tbody');
    // 테이블 행 추가
    // [주의]: data.컬럼명의 '컬럼명'은 collectDataFromModal() 함수의 컬럼명과 일치시킬 것.
    var newRow = `tr>
        <input type="hidden" class="wIdModal" value="${data.wIdModal}"> // 작업번호
        <input type="hidden" class="wStartTimeModal" value="${data.wStartTimeModal}"> // 작업시작시간
        <input type="hidden" class="wEndTimeModal" value="${data.wEndTimeModal}"> // 작업종료시간
        <td class="empIdModal">${data.empIdModal}</td> // 사원번호
        <td onclick="removeRow(this);"
            style="cursor:pointer;
            color:#c82333;
            text-align:center;">
            삭제
        </td>
        </tr>`;
    table.append(newRow);
}
```

추가된 작업자 목록의 데이터를 수집하는 함수

```
/* 등록 - 데이터베이스 저장 로직 */
function workerInsertArr(arrays) {
    $.ajax({
        url: '/workerInsertArr', // 데이터 저장을 처리하는 서버의 API 경로
        type: 'POST',
        contentType: 'application/json',
        data: JSON.stringify(arrays),
        success: function(response) {
            alert('등록이 완료되었습니다.');
            window.location.href = '/workerInsert'; // 저장 후, '/work'로 리다이렉트
        },
        error: function(xhr, status, error) {
            console.error('작업자등록 Error', error);
        }
    });
}
```

공정관리 - 작업자관리 - 등록 / 상세 및 수정, 삭제 (4) - 삭제

7. 등록된 작업자 목록 중 삭제 버튼 클릭 시

등록된 작업자 목록				
부서명	사원번호	사원명	직급	삭제
생산 1팀	100042	박서진	사원	삭제
생산 1팀	100046	정하은	사원	삭제
생산 1팀	100054	이도현	사원	삭제

[WorkController]

```
@PostMapping("/deleteWorker")
@ResponseBody
public ResponseEntity<?> deleteWorker(@RequestParam int emp_id, @RequestParam int work_id) {
    try {
        // 작업자 삭제 로직 실행
        workService.workerInsertModalDeleteWorker(emp_id, work_id);
        return ResponseEntity.ok(Map.of( k1: "success", v1: true,
                                         k2: "message", v2: "작업자가 성공적으로 삭제되었습니다."));
    } catch (Exception e) {
        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR)
            .body(Map.of( k1: "success", v1: false,
                          k2: "message", v2: "삭제 실패: " + e.getMessage()));
    }
}
```

8. 작업자 삭제 confirm 창이 뜨고 작업자 삭제가 된다.

[JavaScript 코드]

```
/* 작업자 삭제를 위한 ajax 코드 */
$(document).on('click', '.delete-button', function() {
    var empId = $(this).data('emp-id');
    var workId = $(this).data('work-id');

    if (confirm('이 작업자를 삭제하시겠습니까?')) {
        $.ajax({
            url: '/deleteWorker', // 서버의 삭제 처리 경로
            type: 'POST',
            data: {
                emp_id: empId,
                work_id: workId
            },
            success: function(response) {
                if (response.success) {
                    alert('작업자가 성공적으로 삭제되었습니다.');
                    $(this).closest('tr').remove(); // 테이블에서 해당 행 삭제
                } else {
                    alert('작업자 삭제에 실패했습니다: ' + response.message);
                }
            }.bind(this), // 현재 클릭한 버튼에 대한 참조를 유지
            error: function(xhr, status, error) {
                alert('삭제 과정에서 오류가 발생했습니다: ' + error);
            }
        });
    }
}); // /. delete-worker ajax
}); // /. delete-button
```

느낀 점(배지현)

[좋았던 점]

- 깃의 활용도를 높여 커밋, 푸시, 풀을 자주 진행함. 이를 통해 깃에 대한 이해도를 높이고 진행도 확인이 가능했음.
- 화면설계와 DB 설계는 여러 번의 회의를 거쳐 탄탄하게 해놓아 크게 고치지 않고 개발을 진행할 수 있었음.
- 팀프로젝트에서는 소통이 굉장히 중요한데 화면설계를 통해 모든 팀원들의 프로젝트 이해도가 높아졌음.
- 팀원들 모두 소통과 연락이 잘 되었으며 개인역량이 뛰어나 피드백도 빠르게 수용하여 프로젝트의 완성도를 높일 수 있어 좋았음.

[아쉬웠던 점]

- 작업관리와 작업자관리를 더 효율적으로 개발할 수 있었을텐데 기간이 짧아 시간을 더 투자할 수 없던 점과 MES 시스템을 만들 때 POP시스템이 필요한데 POP시스템에 대한 부분을 자세히 구현하지 못한 점이 아쉬웠음.
- 기간이 널널하게 주어졌다면 이보다 더 좋은 결과물이 나올 수 있었을 거라는 아쉬움이 있음.

—
김다은

설비관리, 설비점검

설비관리

- 설비 정보 관리
 - 설비 정보 등록, 수정
- 설비 관리
 - 설비 수리
 - 설비 수리 완료
- 설비 수리 이력 관리
 - 설비 수리 내역, 수리 날짜 조회
- 설비 가동 현황
 - 설비 가동 현황 조회

설비점검

- 설비 체크리스트 기준 관리
 - 설비별 체크리스트 정보 등록
 - 체크리스트 조회
- 설비 체크리스트 작성

설비정보관리 & 설비관리체크기준관리 : 모달 등록

설비별 체크리스트 정보 추가

설비번호	<input type="text"/>
설비관리유형	<input type="text"/>
점검방법	<input type="text"/>
조건	<input type="text"/>

등록 **취소**

설비관리유형 **유형** 점검방법 **점검 방법** 체크리스트 조건 **검색** **추가**

✓ **Java Script**
ajax를 사용해 비동기 방식으로 데이터를 저장하고
테이블 행 추가

```
$ajax({
    type: 'POST',
    url: '/machineCheckInfo', // 서버의 컨트롤러 매핑 주소
    data : JSON.stringify(data),
    contentType : "application/json; charset=utf-8",
    success: function(response) {
        // 성공 시 처리
        alert('등록되었습니다.');
        // 새로운 데이터 행 추가
        var newRow = "<tr>" +
            "<td>" + response.mci_id + "</td>" +
            "<td>" + response.modalAddTypeCheck + "</td>" +
            "<td>" + response.modalAddMethod + "</td>" +
            "<td>" + response.modalAddCondition + "</td>" +
            "</tr>";
        $(".tbl-content").append(newRow);
    }
});
```

✓ **Controller**
DB에 텍스트 데이터를 저장한다
DB에 추가된 데이터를 가져온다
VO에 데이터를 넣어준다

```
@PostMapping("/machineInfoAdd")
@ResponseBody
public MachineVO insertMachine(
    @RequestBody MachineAddVO machineAddVO,
    RedirectAttributes rttr
) {
    // 실제 DB에 텍스트 데이터 저장
    // machine_info insert
    int rtn = machineInfoService.insertMachine(machineAddVO);
    rttr.addFlashAttribute(attributeName: "insertSuccessCount", rtn);

    // machine_work insert
    int rtr = machineInfoService.insertMachine2(machineAddVO);
    rttr.addFlashAttribute(attributeName: "insertSuccessCount", rtr);

    // DB에서 입력했던 정보를 바탕으로 DB에서 추가한 설비정보 데이터를 가져온다
    MachineVO machineVOOne = machineInfoService.getMachineOne();

    // 위에서 가져온 데이터를 아래 vo에 맞게 set을 한다.
    MachineVO machineVO = new MachineVO();
    machineVO.setRownum(machineVOOne.getRownum());
    machineVO.setMi_id(machineVOOne.getMi_id());
    machineVO.setMi_name(machineVOOne.getMi_name());
    machineVO.setMi_type(machineVOOne.getMi_type());
    machineVO.setMi_position(machineVOOne.getMi_position());

    return machineVO;
}
```

✓ **Java Script**
모달창에 입력된 값을 객체로 생성

```
$('#insertMachineCheck').on('click', function() {
    var modalAddNumber = $('#modalAddNumber').val();
    var modalAddTypeCheck = $('#modalAddTypeCheck').val();
    var modalAddMethod = $('#modalAddMethod').val();
    var modalAddCondition = $('#modalAddCondition').val();

    // 입력된 값을 객체로 생성
    var data = {
        modalAddNumber: modalAddNumber,
        modalAddTypeCheck: modalAddTypeCheck,
        modalAddMethod: modalAddMethod,
        modalAddCondition: modalAddCondition
    };
});
```

설비정보관리 : 모달 수정

✓ 수정 버튼을 클릭하면 모달창이 뜬다

The screenshot shows a table of service information on the left and a modal window on the right. The table has columns for Service Number, Service Name, Service Type, and Service Location, each with a '수정' (Edit) button. The modal window is titled '설비수정' (Service Update). It contains four input fields corresponding to the table columns, with values '6000001', '재단기1호', '재단기', and '1-3 1' respectively. Below the inputs are two buttons: '등록' (Register) in blue and '취소' (Cancel) in red.

✓ Java Script
해당 행의 데이터를 가져온다

```
$document.ready(function() {  
    // 수정 버튼 클릭 시  
    $document.on('click', '#updateMachine', function() {  
  
        // 해당 행의 데이터 가져오기  
        var row = $(this).closest('tr');  
        var mi_id = row.find("#machineId").text();  
        var mi_name = row.find("#machineName").text();  
        var mi_type = row.find("#machineType").text();  
        var mi_position = row.find("#machinePosition").text();  
  
        // 모달 내의 각 input 요소에 데이터 설정  
        $('#modalUpdateNumber').val(mi_id);  
        $('#modalUpdateName').val(mi_name);  
        $('#modalUpdateType').val(mi_type);  
        $('#modalUpdatePosition').val(mi_position);  
  
        // 모달 열기  
        $('#updateModal').modal('show');  
    });  
});
```

설비체크기준관리

✓ 체크리스트 등록 : MyBatis SQL 쿼리문

select문을 사용해, 입력한 설비관리유형, 점검방법, 조건과 같은 값이 DB에 없으면 시퀀스를, 있으면 같은 체크리스트 ID를 insert한다

```
<insert id="addMachineCheck">
    INSERT INTO MACHINE_CHECK_INFO (mci_id, mci_mi_id, mci_div, mci_method, mci_condition)
    SELECT
        CASE
            WHEN EXISTS (
                SELECT 1
                FROM MACHINE_CHECK_INFO
                WHERE mci_div = #{modalAddTypeCheck}
                AND mci_method = #{modalAddMethod}
                AND mci_condition = #{modalAddCondition}
            ) THEN (
                SELECT MAX(mci_id)
                FROM MACHINE_CHECK_INFO
                WHERE mci_div = #{modalAddTypeCheck}
                AND mci_method = #{modalAddMethod}
                AND mci_condition = #{modalAddCondition}
            )
            ELSE MACHINE_CHECK_INFO_SEQ.NEXTVAL
        END,
        #{modalAddNumber},
        #{modalAddTypeCheck},
        #{modalAddMethod},
        #{modalAddCondition}
    FROM dual
</insert>
```

유형	체크리스트	진동	전기	외관	소음	부품	금유	온도
80000								

✓ 체크리스트 검색 : jstl

forEach를 사용해 DB에서 설비관리 유형을 가져와 보여준다

```
<select id="find_machine_check_type" name="find_machine_check_type" class="find_machine_check_type">
    <option class="checkType" disabled selected>유형</option>
    <c:forEach var="list" items = "${getMachineCheckType}">
        <option value="${list.mci_div}" ${list.mci_div == searchMachineCheckVO.find_machine_check_type ? 'selected' : ''}>
            ${list.mci_div}</option>
    </c:forEach>
</select>
```

설비 체크리스트 작성

설비명	재단기1호	작성자	100001	작성일	연도-월-일	<input type="button" value="검색"/>
-----	-------	-----	--------	-----	--------	-----------------------------------

```
@GetMapping("/machineCheck")
public String machineCheck(SearchMachineVO searchMachineVO, Model model) {
    // 설비 ID와 이름 가져오기
    List<MachineVO> getMachineInfo = machineMNGService.getMachineInfo();
    model.addAttribute("getMachineInfo", getMachineInfo);

    // 리스트 가져오기
    List<MachineCheckVO> getMachineCheckAll;
    if (searchMachineVO != null && searchMachineVO.getFind_machine_id() != null) {
        getMachineCheckAll = machineMNGService.getMachineCheckAll(searchMachineVO);
    } else {
        // 선택된 설비가 없는 경우 빈 리스트 반환
        getMachineCheckAll = new ArrayList<>();
    }
    model.addAttribute("getMachineCheckAll", getMachineCheckAll);

    return "machine/machineCheck";
}
```

✓ Controller

체크리스트 답변에 따라 업데이트를 실행한다

NO	설비관리유형	점검방법	체크리스트 조건	Y / N	비고	저장
1	전기	육안	누적이 있는가	<input checked="" type="radio"/> O/O	<input type="text"/>	저장

```
@PostMapping("/machineCheckAdd")
@ResponseBody
public ResponseEntity<MachineCheckRecordVO> machineCheckRecord(@RequestBody MachineCheckRecordVO machineCheckRecordVO,
    RedirectAttributes rttr) {
    // 체크리스트 기록 추가
    int rtn = machineMNGService.insertMachineCheckRecord(machineCheckRecordVO);
    rttr.addFlashAttribute("insertSuccessCount", rtn);

    // 체크리스트 답변에 따라 업데이트 실행
    String newCondition = machineCheckRecordVO.getMcr_answer();
    int newMI_ID = machineCheckRecordVO.getMcr_mi_id();

    if (newCondition.equals("Y")) {
        // 설비 상태와 설비 가동 현황 업데이트
        String newStatus = machineCheckRecordVO.getMcr_answer();
        int updateSuccessCount = machineMNGService.updateMachineCondition(newCondition, newStatus, newMI_ID);
        rttr.addFlashAttribute("updateSuccessCount", updateSuccessCount);
    } else if (newCondition.equals("N")) {
        // 설비 상태 업데이트
        int updateMcSuccessCount = machineMNGService.updateMcWork(newCondition, newMI_ID);
        rttr.addFlashAttribute("updateSuccessCount", updateMcSuccessCount);
    }

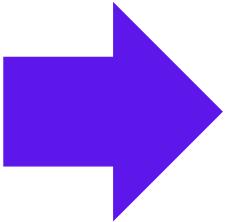
    return ResponseEntity.ok(machineCheckRecordVO);
}
```

✓ Controller

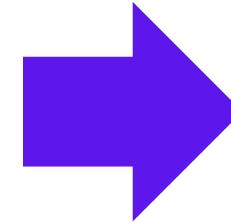
선택한 설비명의 설비 ID와 같은 체크리스트를 보여준다
선택된 설비가 없는 경우 빈 리스트를 반환한다

설비 관리

상태	비고
수리요청	수리



상태	비고
수리중	수리완료



상태	비고
수리완료	

```
$(document).ready(function() {
    // 각 수리 버튼에 대한 클릭 이벤트 처리
    $('.repair-button').click(function() {
        // 해당 설비의 고유 식별자 가져오기
        var mwMiId = $(this).data('mw-mi-id');

        // 콘솔 확인
        console.log('수리 버튼이 클릭되었습니다. 설비 ID:', mwMiId);

        // 서버로 데이터를 전송하여 상태를 업데이트하는 코드
        $.ajax({
            type: "POST",
            url: "/updateRepairStatus", // 서버의 업데이트 처리를 담당하는 URL
            data: JSON.stringify({ mw_mi_id: mwMiId }),
            contentType : "application/json; charset=utf-8",
            success: function(response) {
                alert('수리중으로 되었습니다.');
                console.log('수리 상태가 업데이트되었습니다.');
                // 페이지 새로고침
                window.location.reload();
            },
            error: function(xhr, status, error) {
                // 오류 발생 시 처리하는 코드
                console.error("상태 업데이트 중 오류 발생:", error);
                alert('등록에 실패하였습니다.');
            }
        });
    });
});
```

```
$(document).ready(function() {
    // 각 수리 완료 버튼에 대한 클릭 이벤트 처리
    $('.repairCompleted').click(function() {
        // 해당 설비의 고유 식별자 가져오기
        var mwMiId = $(this).data('mw-mi-id');

        // 콘솔 확인
        console.log('수리완료 버튼이 클릭되었습니다. 설비 ID:', mwMiId);

        // 서버로 데이터를 전송하여 상태를 업데이트하는 코드
        $.ajax({
            type: "POST",
            url: "/mcRepairCompleted", // 서버의 업데이트 처리를 담당하는 URL
            data: JSON.stringify({ mw_mi_id: mwMiId }),
            contentType : "application/json; charset=utf-8",
            success: function(response) {
                alert('수리완료 되었습니다.');
                console.log('수리 상태가 업데이트되었습니다.');
                // 페이지 새로고침
                window.location.reload();
            },
            error: function(xhr, status, error) {
                // 오류 발생 시 처리하는 코드
                console.error("상태 업데이트 중 오류 발생:", error);
                alert('등록에 실패하였습니다.');
            }
        });
    });
});
```

```
@PostMapping("/mcRepairCompleted")
@ResponseBody
public String mcRepairCompleted(@RequestBody MachineWorkVO machineWorkVO, MachineRepairAddVO machineRepairAddVO,
                                 RedirectAttributes rttr){
    int mwMiId = machineWorkVO.getMw_mi_id();

    // 설비 관리 수리완료 버튼 클릭 시 '수리완료'로 업데이트
    int rtr = machineInfoService.mcRepairCompleted(machineWorkVO);
    rttr.addFlashAttribute(attributeName: "updateSuccessCount", rtr);

    // 수리완료 되었을 때 설비수리이력 테이블에 데이터 insert
    machineRepairAddVO = new MachineRepairAddVO();
    machineRepairAddVO.setMw_mi_id(mwMiId); // 업데이트 된 설비 번호 가져오기 & 사용
    int rtt = machineInfoService.addMcRepair(machineRepairAddVO);
    rttr.addFlashAttribute(attributeName: "insertSuccessCount", rtt);

    return "redirect:/machineManagement";
}
```



Controller

버튼 클릭 시, 상태를 업데이트한다

수리완료 되었을 때, 설비수리 이력 테이블에 데이터를 insert한다

✓ Java Script

각 버튼 클릭 시,

ajax를 사용해 서버로 데이터를 전송하여 상태를 업데이트 한다

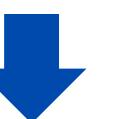
설비수리이력조회 & 설비가동 현황

NO	설비번호	설비명	설비유형	설비위치	수리이력내용	수리날짜
1	6000001	재단기1호	재단기	1-3 1	전선 손상	2024-04-22

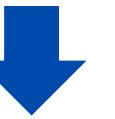
✓ 설비수리이력조회

설비관리에서 수리완료를 클릭하면
수리 날짜와 수리 이력 내용이 뜬다

NO	설비번호	설비명	설비유형	설비위치	작동시작시간	작동종료시간	설비상태	설비가동현황
1	6000001	재단기1호	재단기	1-3 1			수리요청	비가동



NO	설비번호	설비명	설비유형	설비위치	작동시작시간	작동종료시간	설비상태	설비가동현황
1	6000001	재단기1호	재단기	1-3 1			수리중	수리중



NO	설비번호	설비명	설비유형	설비위치	작동시작시간	작동종료시간	설비상태	설비가동현황
1	6000001	재단기1호	재단기	1-3 1			수리완료	비가동

✓ 설비가동 현황

설비관리에서 클릭하는 버튼에 따라
상태와 현황이 바뀐다

✓ 설비가동 현황 MyBatis SQL 쿼리문

```
SELECT
    ROWNUM,
    mw.*,
FROM (
    SELECT
        MW.MW_MI_ID,
        MI.MI_NAME,
        MI.MI_TYPE,
        PI.PI_SEQ,
        MI.MI_POSITION,
        MAX(WT.W_START_TIME) AS W_START_TIME,
        MAX(WT.W_END_TIME) AS W_END_TIME,
        MW.MW_CONDITION,
        MW.MW_STATUS
    FROM
        MACHINE_WORK MW
    JOIN
        MACHINE_INFO MI ON MW.MW_MI_ID = MI.MI_ID
    LEFT JOIN
        PROCESS_INFO PI ON MW.MW_PI_ID = PI.PI_MACHINE_ID
    LEFT JOIN (
        SELECT
            W.W_PI_ID,
            W.W_START_TIME,
            W.W_END_TIME
        FROM
            WORK W
    ) WT ON PI.PI_ID = WT.W_PI_ID
    GROUP BY
        MW.MW_MI_ID,
        MI.MI_NAME,
        MI.MI_TYPE,
        PI.PI_SEQ,
        MI.MI_POSITION,
        MW.MW_CONDITION,
        MW.MW_STATUS
    ORDER BY
        MW.MW_MI_ID
) mw
where 1=1
<if test="find_machine_name != null and find_machine_name != ''">
    AND mi_name Like '%' || #{find_machine_name} || '%'
</if>
<if test="find_machine_type != null and find_machine_type != ''">
    AND mi_type Like '%' || #{find_machine_type} || '%'
</if>
<if test="find_machine_status != null and find_machine_status != ''">
    AND mw_status Like '%' || #{find_machine_status} || '%'
</if>
```

느낀 점(김다은)

[좋았던 점]

- 프로젝트 설계를 탄탄하게 하여, 각자 프로젝트에 대한 이해도를 높이고 개발을 원활하게 진행하며 효율적으로 프로젝트를 수행할 수 있어서 좋았음.
- 팀원들간 의사소통이 잘 되어서 수정 사항이 바로 진행되고, 서로의 보완점이 되어주어 좋았음.
- 다양한 코드와 문제 해결 방식을 보고 배울 수 있어서 좋았음.
- 깃과 깃허브에 대한 이해도를 높이고 활용할 수 있어 좋았음.

[아쉬웠던 점]

- 개발 기간이 더 길었다면 코드를 좀 더 보완할 수 있을 것 같음.

정수하

품질관리, 재고관리

품질관리

1. 수입검사관리
2. 공정검사관리
3. 출하검사관리
 - 등록 시, 특정 내용을 작성하면, 관련된 내용이 자동으로 채워지도록 함
 - 동일한 대상의 검사여도 불량유형이 다양하므로,
 - 그에 따른 [누적불량수량]과 [양품수량]을 계산하도록 함
 - 등록 시, 트랜잭션을 사용하여, [품질검사기록]은 작성한 수 만큼 insert하고, 필요한 나머지 테이블은 1번만 insert 또는 update 되도록 함
4. 공정불량실적

재고관리

1. 제품재고조회

품질관리 - 수입검사관리

- 검색 및 조회
- 등록
- 수정

새로고침 ←

↻ 검색

검색

☰ 수입검사현황

⊕ 자재불량등록

등록

수정

NO	자재번호	자재명	자재용도	계약번호	거래처명	단위	계약입고수량	검사수량 (입고수량)	불량유형1	불량유형2	불량수량	불량률(%)	양품수량	검사일자 (입고일자)	비고	수정
1	20,000,001	소나무	원자재	1,000,031	나무마켓	m	10	10	손상	균열(갈라짐)	1	10.000	8	2024-04-15	20cm 깊이, 1m의 균열	수정
2	20,000,001	소나무	원자재	1,000,031	나무마켓	m	10	10	손상	파손	1	10.000	8	2024-04-15		수정
3	20,000,022	참나무	원자재	1,000,032	원목킹	m	10	10	형태변형	꺼짐(찌그러짐)	1	10.000	9	2024-04-15		수정
4	20,000,025	자작나무	원자재	1,000,033	소나무가게	m	10	10	기타		0	0.000	10	2024-04-16	불량없음	수정

품질관리 - 수입검사관리

• 검색 및 조회

수입검사관리

자재번호	계약번호					
불량유형	불량유형1 선택	불량유형2 선택	검사일자 (입고일자)	연도-월-일	연도-월-일	검색

새로고침 ← 검색

[검색용 데이터를 조회]

- 자재번호
- 계약번호
- ▶ **datalist 목록으로 조회하면서 입력 가능**

자재번호	<input type="text"/>
<ul style="list-style-type: none">20000001200000022000000320000004200000052000000620000007200000082000000920000010200000112000001220000013	

[jsp 코드]

datalist 목록으로 조회하면서 입력 가능

```
<!-- 자재번호 -->
<div class="col-sm-1 inspection-name">자재번호</div>
<div class="col-sm-3 inspection-text">
    <input type="search" list="matIDList" class="col-sm-12" id="matID" name="matID" value="${cri.matID}">
    <datalist id="matIDList">
        <c:forEach items="${matIDList}" var="mat">
            <option value="${mat.matID}">
                <c:if test="${cri.matID == mat.matID}">selected</c:if> ${mat.matID}
            </option>
        </c:forEach>
    </datalist>
</div><!-- /.자재번호 -->
```

[Controller 코드]

검색용 데이터를 로드하는 메소드

```
/*
 * Desc: 검색장 - 검색용 데이터를 로드하는 메소드.
 */
1 usage
private void loadSearchData(CriteriaInspIBVO cri, Model model) {
    model.addAttribute(attributeName: "contractIDList", inspectionIBService.getContractIDList()); // 계약ID(검색용)
    model.addAttribute(attributeName: "matIDList", inspectionIBService.getMatIDList()); // 자재ID(검색용)
    model.addAttribute(attributeName: "qsDiv1List", inspectionIBService.getQsDiv1List()); // 불량유형1(검색용)
    model.addAttribute(attributeName: "qsDiv2List", loadQsDiv2List(cri.getQsDiv1())); // [불량유형1]에 따른 [불량유형2], (검색용)
}
```

품질관리 - 수입검사관리

• 검색 및 조회

수입검사관리

자재번호	계약번호				
불량유형	불량유형1 선택 불량유형2 선택	검사일자 (입고일자)	연도-월-일	연도-월-일	검색

새로고침 ← 검색

[불량유형1]에 따른 [불량유형2] option 조회

[불량유형1] 선택 전

불량유형1 선택 불량유형2 선택

불량유형1 선택 불량유형2 선택

- 손상
- 형태변형
- 도장불량
- 조립불량
- 수치불량
- 기타



[불량유형1] 선택 후

손상 균열(갈라짐)

균열(갈라짐)

- 균열(갈라짐)
- 파손
- 찌힘
- 흠집

[JavaScript 코드]

```
if (qsDiv1) {
    $.ajax({
        url: '/qsDiv2List?qsDiv1=' + qsDiv1,
        type: 'GET',
        dataType: 'json',
        success: function(data) {
            console.log('response data', data);

            // 데이터가 비어있지 않으면 옵션 추가
            if (data.length > 0) {
                $.each(data, function(index, item) {
                    qsDiv2Select.append($('', {
                        value: item.qsDiv2,
                        text: item.qsDiv2
                    }));
                });
            } else {
                qsDiv2Select.append('<option value="">No Options Available</option>');
            }
        },
        error: function(xhr, status, error) {
            console.error("Error loading qsDiv2 options: ", error);
        }
    });
}
```

ajax 사용하여

[불량유형1] 선택 시, 그것에 해당하는 [불량유형2]의 값을 불러옴

품질관리 - 수입검사관리

● 검색 및 조회

자재번호 20000001

검색

[키워드] 입력 후, [검색] 버튼 누르면 해당 내용이 조회됨

예시: [자재번호] 입력 후, 검색

[검색 전]

모든 내역 조회

수입검사현황																	
NO	자재번호	자재명	자재용도	계약번호	거래처명	단위	계약입고수량	검사수량 (입고수량)	불량유형1	불량수량	불량률(%)	양품수량	검사일자 (입고일자)	비고	수정		
1	20,000,001	소나무	원자재	1,000,031	나무마켓	m	10	10	손상	균열(길라짐)	1	10.000	8	2024-04-15	20cm 길이, 1m의 균열	수정	
2	20,000,001	소나무	원자재	1,000,031	나무마켓	m	10	10	손상	파손	1	10.000	8	2024-04-15		수정	
3	20,000,022	침나무	원자재	1,000,032	원목킹	m	10	10	형태변형	꺼짐(피그러짐)	1	10.000	9	2024-04-15		수정	
4	20,000,025	자작나무	원자재	1,000,033	소나무가게	m	10	10	기타		0	0.000	10	2024-04-16	불량없음	수정	

[검색 후]

해당하는 내용만 조회

수입검사현황						
NO	자재번호	자재명	자재용도	계약번호	거래처명	단위
1	20,000,001	소나무	원자재	1,000,031	나무마켓	m
2	20,000,001	소나무	원자재	1,000,031	나무마켓	m

[Controller 코드]

```
/**  
 * Desc: 품질관리- 수입검사관리(자재IB) - 조회/  
 * 검색조건에 따라 다른 결과를 모델에 추가하고, 결과 페이지를 렌더링한다.  
 * @param cri: 검색 조건을 담은 VO 객체  
 * @param model: 뷰에 데이터를 전달하기 위한 모델  
 * @return: qualityInspection/inspectionIB  
 */  
no usages  
@GetMapping("/inspectionIB")  
public String getInspectionIBList(CriteriaInspIBVO cri, Model model) {  
    log.info("수입검사관리 페이지");  
  
    loadSearchData(cri, model); // 검색용 데이터 로드  
    loadModalData(cri, model); // 모달용 데이터 로드  
  
    // 수입검사현황 목록 조회  
    List<InspIBListVO> inspectionIBList = inspectionIBService.getInspectionIBList(cri);  
    if (inspectionIBList.isEmpty()) {  
        log.warn("검색 결과가 없습니다.");  
        /*model.addAttribute("message", "검색된 결과가 없습니다.");*/  
    } else {  
        model.addAttribute(attributeName: "inspIBList", inspectionIBList);  
        model.addAttribute(attributeName: "cri", cri);  
    }  
  
    return "qualityInspection/inspectionIB"; // View 반환  
}
```

품질관리 - 수입검사관리

● 등록

[수입검사] - 등록 모달창

[자재불량등록] 버튼 클릭 시, 등록 모달 뜹

수입검사 - 자재불량 및 입고등록

계약번호		거래처명		단위	
자재번호		자재명		자재용도	
계약입고수량		검사수량 (입고수량)		불량수량	
불량유형1	불량유형1 선택	불량유형2			
비고					

[추가] 버튼 클릭 시, 목록에 추가됨

☰ 자재불량목록

NO	자재번호	자재명	자재용도	계약번호	거래처명	단위	계약입고수량	검사수량 (입고수량)	불량유형1	불량유형2	불량수량	불량률(%)	양품수량
1	20000001	소나무	원자재	1000031	나무마켓	m	10	10	손상	균열(갈라짐)	1	10.00%	7
2	20000001	소나무	원자재	1000031	나무마켓	m	10	10	형태변형	꺼짐(찌그러짐)	1	10.00%	7
3	20000001	소나무	원자재	1000031	나무마켓	m	10	10	손상	흡집	1	10.00%	7

등록 취소

[등록 모달창] 내의 로직

- 불량 정보 입력 후, [추가] 버튼 클릭
- 입력한 내용을 [자재불량목록] 테이블에서 확인 가능
- 원하는 만큼 추가한 후, [등록] 버튼 클릭
- 테이블에서 데이터를 수집한 후, ajax 통신으로 DB 저장

동일한 검사 대상이어도,
다양한 불량이라면,

자동으로 양품수량을 계산하여 표기됨

품질관리 - 수입검사관리

● 등록

[계약번호] 입력 시, 관련 내용이 자동으로 채워짐

[계약번호] 입력 전

계약번호	<input type="text"/>	거래처명	<input type="text"/>	단위	<input type="text"/>
자재번호	<input type="text"/> 1000031	자재명	<input type="text"/>	자재용도	<input type="text"/>
계약입고수량	<input type="text"/> 1000033	검사수량 (입고수량)	<input type="text"/>	불량수량	<input type="text"/>
불량유형1	불량유형1 선택	불량유형2	불량유형2 선택		
비고	<input type="text"/>				

[계약번호] 입력 후

계약번호	<input type="text"/> 1000031	거래처명	<input type="text"/> 나무마켓	단위	<input type="text"/> m
자재번호	<input type="text"/> 20000001	자재명	<input type="text"/> 소나무	자재용도	<input type="text"/> 원자재
계약입고수량	<input type="text"/> 10	검사수량 (입고수량)	<input type="text"/>	불량수량	<input type="text"/>
불량유형1	불량유형1 선택	불량유형2	불량유형2 선택		
비고	<input type="text"/>				

[JavaScript 코드]

```
/* 모달창 - [계약번호] 입력에 따른 [거래처명], [계약입고수량], [단위], [자재번호], [자재명], [자재용도] 자동 채우기 */
$('#contractIDModal').change(function() {
    var contractID = $(this).val();

    $.ajax({
        url: '/contractDetailsModal', // 이 URL은 Backend에서 처리할 경로
        type: 'GET',
        data: {contractIDModal: contractID},

        success: function(data) { // 성공 시
            $('#companyNameModal').val(data.companyNameModal);
            $('#contractQuantityModal').val(data.contractQuantityModal);
            $('#unitModal').val(data.unitModal);
            $('#matIDModal').val(data.matIDModal);
            $('#matNameModal').val(data.matNameModal);
            $('#matUsesModal').val(data.matUsesModal);
        },

        error: function(xhr, status, error) { // 실패 시
            console.log("Error contract details: ", error);
        }
    });
});
```

ajax 사용하여,
[계약번호] 입력 시, 관련 내용이 자동으로 채워지도록 함

품질관리 - 수입검사관리

● 등록

[불량검사목록] 추가

[불량검사목록] 추가 전

계약번호	1000031	거래처명	나무마켓	단위	m
자재번호	20000001	자재명	소나무	자재용도	원자재
계약입고수량	10	검사수량 (입고수량)	10	불량수량	1
불량유형1	손상	불량유형2	균열(갈라짐)		
비고					

[1] [추가] 버튼을 누르면

[불량검사목록] 추가 후

계약번호		거래처명		단위	
자재번호		자재명		자재용도	
계약입고수량		검사수량 (입고수량)		불량수량	
불량유형1	불량유형1 선택	불량유형2			
비고					

[2] 입력값이 초기화되고,

자재불량목록														
NO	자재번호	자재명	자재용도	계약번호	거래처명	단위	계약입고수량	검사수량 (입고수량)	불량유형1	불량유형2	불량수량	불량률(%)	양풀수량	등록일
1	20000001	소나무	원자재	1000031	나무마켓	m	10	10	손상	균열(갈라짐)	1	10.00%	9	2023-09-15

[3] 목록이 보임

[JavaScript 코드]

```
/* 모달창 - [추가] 버튼 클릭 이벤트 */
$('#addInspectionBtn').click(function() {
    console.log('추가 버튼 클릭'); // 디버깅을 위한 코드
    // 모달창에서 입력된 데이터 수집
    var data = collectDataFromModal();
    console.log('모달창에서 입력된 데이터: ', data);

    // 입력 필드 유효성 검사
    if(!validateFields(data)) {
        return; // 유효성 검사 실패 시 추가 작업 중단
    }
    console.log('유효성 검사 데이터: ', data); // 유효성 검사 data 디버깅

    // 데이터 목록에 추가
    addToInspectionList(data);

    // [자재불량목록]을 보이게 한다.
    /*document.querySelector('.newInspList').style.display = 'block';*/
    $('.newInspList').css('display', 'block');
    console.log('display = block');

    // 입력 필드 초기화
    clearInputFields();
    console.log('입력필드 초기화');
});
```

품질관리 - 수입검사관리

● 등록

[불량검사목록] 추가 시, [양품수량] 계산

자재불량목록													
NO	자재번호	자재명	자재용도	계약번호	거래처명	단위	계약입고수량	검사수량 (입고수량)	불량유형1	불량유형2	불량수량	불량률(%)	양품수량
1	20000001	소나무	원자재	1000031	나무마켓	m	10	10	손상	균열(갈라짐)	1	10.00%	7
2	20000001	소나무	원자재	1000031	나무마켓	m	10	10	형태변형	꺼짐(찌그러짐)	1	10.00%	7
3	20000001	소나무	원자재	1000031	나무마켓	m	10	10	손상	흠집	1	10.00%	7

동일한 검사 대상이어도,
다양한 불량이라면,

자동으로 양품수량을 계산하여 표기됨

[JavaScript 코드]

누적 불량수량 계산 함수

```
/* 누적 불량수량 계산 함수 */
// 동일한 [계약번호 & 자재번호]이지만, 다른 종류의 불량인 경우를 위해, 계산이 필요함.
function totalDefectQuantity(data) {
    var total = 0;
    $('.newInspList tbody tr').each(function() {
        if($(this).find('.contractID').text() === data.contractID
            && data.materialID === $(this).find('.materialID').text()) {
            total += parseInt($(this).find('.poorQuantity').text(), 10);
        }
    });
    return total;
}
```

양품수량 계산 함수

```
function updateGoodQuantities() {
    $('.newInspList tbody tr').each(function() {
        var row = $(this);
        var inspectionQuantity = parseInt(row.find('.inspectionQuantity').text(), 10); // 검사수량
        var totalDefects = totalDefectQuantity({
            contractID: row.find('.contractID').text(),
            materialID: row.find('.materialID').text()
        });
        var goodQuantity = Math.max(inspectionQuantity - totalDefects, 0); // 음수 방지
        row.find('.goodQuantity').text(goodQuantity); // 목록의 양품수량 Update
    });
}
```

목록이 추가/삭제 될 때마다, [양품수량]을 최신화하는 함수

```
/* 양품수량 계산 및 테이블 목록에 업데이트하는 함수 */
// 동일한 [계약번호 & 자재번호]이지만, 다른 종류의 불량인 경우를 위해, 계산이 필요함.
function updateGoodQuantities() {
    $('.newInspList tbody tr').each(function() {
        var row = $(this);
        var inspectionQuantity = parseInt(row.find('.inspectionQuantity').text(), 10); // 검사수량
        var totalDefects = totalDefectQuantity({
            contractID: row.find('.contractID').text(),
            materialID: row.find('.materialID').text()
        });
        var goodQuantity = Math.max(inspectionQuantity - totalDefects, 0); // 음수 방지
        row.find('.goodQuantity').text(goodQuantity); // 목록의 양품수량 Update
    });
}
```

품질관리 - 수입검사관리

● 등록

[등록] 버튼 클릭 시, DB에 저장

NO	자재번호	자재명	자재용도	계약번호	거래처명	단위	계약입고수량	검사수량 (입고수량)	불량유형1	불량유형2	불량수량	불량률(%)	양품수량	
1	20000001	소나무	원자재	1000031	나무마켓	m	10	10	손상	균열(갈라짐)	1	10.00%	7	
2	20000001	소나무	원자재	1000031	나무마켓	m	10	10	형태변형	끼짐(찌그러짐)	1	10.00%	7	
3	20000001	소나무	원자재	1000031	나무마켓	m	10	10	손상	흡집	1	10.00%	7	

등록 취소

목록의 데이터를 수집하는 함수

```
/* 등록 - 테이블에서 데이터 수집 */
function collectItemsFromTable() {
    var items = [];
    $('.newInspList tbody tr').each(function() {
        var row = $(this);
        items.push({
            matIDModal: row.find('.materialID').text(),
            matNameModal: row.find('.materialName').text(),
            matUsesModal: row.find('.materialUses').text(),
            contractIDModal: row.find('.contractID').text(),
            companyNameModal: row.find('.companyName').text(),
            unitModal: row.find('.unit').text(),
            contractQuantityModal: row.find('.contractQuantity').text(),
            inspectionQuantityModal: row.find('.inspectionQuantity').text(),
            qsDiv1Modal: row.find('.qsDiv1').text(),
            qsDiv2Modal: row.find('.qsDiv2').text() || '',
            poorQuantityModal: row.find('.poorQuantity').text(),
            defectRate: row.find('.defectRate').text(),
            goodQuantityModal: row.find('.goodQuantity').text(),
            inspectionDateModal: row.find('.inspectionDate').text(),
            notesModal: row.find('.notes').text() || ''
        });
    });
    return items;
}
```

[JavaScript 코드]

[등록] 버튼 클릭 시, 실행하는 내용

```
/* 모달창 - [등록] 버튼 클릭 이벤트 */
$('#registerBtn').click(function() {
    var items = collectItemsFromTable(); // 테이블에서 모든 항목 수집
    if(items.length === 0) {
        alert('등록할 내용이 없습니다.');
        return;
    }
    if(confirm('자재불량정보를 등록하시겠습니까?')) {
        registerInspectionItems(items); // 수집된 데이터를 DB에 저장
    }
});
```

DB에 저장하는 함수

```
/* 등록 - 데이터베이스 저장 로직 */
function registerInspectionItems(items) {
    $.ajax({
        url: '/registerInspectionItems', // 데이터 저장을 처리하는 서버의 API 경로
        type: 'POST',
        contentType: 'application/json',
        data: JSON.stringify(items),
        success: function(response) {
            alert('등록이 완료되었습니다.');
            /*window.location.reload(); // 저장 후, 페이지 새로고침*/
            window.location.href = '/inspectionIB'; // 저장 후, '/inspectionIB'로 리다이렉트
        },
        error: function(xhr, status, error) {
            console.error('자재불량등록 Error', error);
        }
    });
}
```

품질관리 - 수입검사관리

● 등록

[ServiceImpl 코드]

```
/*
 * Desc: 자재불량 등록 시, DB 저장 - [품질검사 테이블], [재고 테이블]
 * @return: inspectionIB.js 코드에 '/inspectionIB'로 리다이렉트하는 코드가 있음.
 * [계약(CONTRACT) 테이블]의 업데이트는 같은 계약번호에 대해서는 단 한 번만 업데이트 수행
 * [Set]:
 */
1 usage  suha +1
@Override
@Transactional
public void registerInspectionItems(List<InspIBInsertVO> items) {
    Map<String, Boolean> updatedContracts = new HashMap<>(); // 업데이트된 계약번호와 자재번호 조합을 저장하기 위한 Map(계약 테이블용)
    Map<String, Boolean> updatedInventory = new HashMap<>(); // 업데이트된 계약번호와 자재번호 조합을 저장하기 위한 Map(재고 테이블용)

    for(InspIBInsertVO item: items) {
        // [품질검사 테이블]에 데이터 저장
        inspectionIBMapper.insertQualityInspection(item);

        // 고유한 키 생성 (계약번호와 자재번호의 조합)
        String contractKey = item.getContractIDModal() + "-" + item.getMatIDModal();
        String inventoryKey = item.getContractIDModal() + "-" + item.getMatIDModal();

        // 계약 테이블 업데이트: 이미 업데이트된 조합인지 확인하고, 아니라면 업데이트 수행
        if(!updatedContracts.containsKey(contractKey)) {
            // [계약 테이블] 업데이트
            inspectionIBMapper.updateContract(item);
            updatedContracts.put(contractKey, true); // 업데이트된 조합을 Map에 추가
        }

        // 재고 테이블 업데이트: 이미 업데이트된 조합인지 확인하고, 아니라면 업데이트 수행
        if(!updatedInventory.containsKey(inventoryKey)) {
            // [재고 테이블] 업데이트 또는 삽입
            inspectionIBMapper.insertOrUpdateInventory(item);
            updatedInventory.put(inventoryKey, true); // 업데이트된 조합을 Map에 추가
        }
    }
}
```

트랜잭션 사용하여,
[등록] 버튼 1회 클릭 시, 3개의 테이블에 데이터 저장



[최종 등록] 시, DB 저장 로직

품질 검사 테이블

기존 데이터가 없으므로, INSERT

계약 테이블

기존 데이터가 있으며,
[실제 수량], [실제 입고일] 컬럼에만 값을 넣으므로, UPDATE

재고 테이블

불량검사를 한 '자재(ID)'의 기존 데이터가 없을 경우,
새로운 '자재ID와 양품수량'을 INSERT

기존 데이터가 있을 경우,
해당하는 '자재ID'의 수량에 '기존수량+양품수량'을 UPDATE

[Controller 코드]

```
/*
 * Desc: 자재불량 등록 시, DB 저장 - [품질검사 테이블], [재고 테이블]
 * @param items 수입검사 항목 목록, 클라이언트로부터 JSON 형태로 받아 InspIBInsertVO 객체 리스트로 변환
 * @return 성공 시 HTTP 200 상태와 성공메시지를 반환, 실패 시 HTTP 500 상태와 함께 실패 메시지를 반환
 * ResponseEntity<?>는 Spring Framework에서 HTTP 요청에 대한 응답을 표현하는 클래스다.
 * 이 클래스를 사용하면 HTTP 상태 코드, 응답 본문, 헤더 등을 포함하는 전체 HTTP 응답을 구성하고 관리할 수 있다.
 * ResponseEntity는 제네릭을 사용하여 응답 본문의 타입을 지정할 수 있습니다. 여기서 <?>는 와일드카드를 나타내며,
 * 응답 본문의 타입이 지정되지 않았거나 다양한 타입을 허용할 때 사용된다.
 */
no usages
@PostMapping("/registerInspectionItems")
@ResponseBody
public ResponseEntity<?> registerInspectionItems(
    @RequestBody List<InspIBInsertVO> items
) {
    log.info(items);
    try {
        inspectionIBService.registerInspectionItems(items);
        return ResponseEntity.ok().body(Map.of(k1: "success", v1: true, k2: "message", v2: "등록이 성공되었습니다."));
    } catch (Exception e) {
        log.error("등록 실패, Error: {" + e.getMessage() + e);
        return ResponseEntity.status(HttpStatus.INTERNAL_SERVER_ERROR).body(Map.of(k1: "success", v1: false, k2: "message", v2: "등록에 실패하였습니다. 에러: " + e.getMessage()));
    }
}
```

품질관리 - 공정검사관리

● 조회

[전체 화면]

제조LOT번호 공정번호 공정명
불량유형 불량유형1 불량유형2 작업번호 설비번호

검색 새로고침

등록

수정

NO	작업번호	제조LOT번호	설비번호	공정번호	공정명	단위	일일총작업수량	검사수량	불량수량	불량률(%)	양품수량	불량율(%)
1	2	300,001	6,000,052	1,001	원자재준비	ea	2,500	2,500	10	0.400	2,480	손상
2	2	300,001	6,000,052	1,001	원자재준비	ea	2,500	2,500	7	0.280	2,480	형태
3	2	300,001	6,000,052	1,001	원자재준비	ea	2,500	2,500	3	0.120	2,480	형태
4	1	300,001	6,000,052	1,001	원자재준비	ea	1,888	1,888	2	0.106	1,880	손상
5	1	300,001	6,000,052	1,001	원자재준비	ea	1,888	1,888	6	0.318	1,880	형태

품질관리 - 공정검사관리

● 등록

[등록 모달창]

공정검사 - 공정불량등록

→[작업번호] 입력 시, [공정번호], [공정명], [제조LOT번호], [설비번호], [단위], [일일총작업수량] 자동 채워짐

작업번호	5	공정번호	1011	공정명	적재
제조LOT번호	300001	설비번호	6000053	단위	ea
일일총작업수량	600	검사수량	600	불량수량	1
불량유형1	조립불량 ▼	불량유형2	헐거움 ▼		
비고					

→**(+) 추가** →추가: 하단의 목록 추가

☰ 공정불량목록

NO	작업수량	검사수량	불량수량	불량률(%)	양품수량	불량유형1	불량유형2	검사일자	비고
1	0	600	12	2.00	588	도장불량	변색	2024-04-24	

등록 취소

→등록: DB에 저장

품질관리 - 공정검사관리

- 수정

[수정 모달창]

공정검사 - 공정불량수정

→ 해당하는 행의 데이터를 담은 채로 수정 모달창 열림

작업번호	1	공정번호	1001	공정명	원자재준비
제조LOT번호	300001	설비번호	6000052	단위	ea
일일총작업수량	1888	검사수량	1888	불량수량	6
불량유형1	형태변형 ▼	불량유형2	꺼짐(찌그 ▼)		
비고					

→ 수정 가능한 부분

등록 취소

→ 등록: 수정사항을 DB에 저장

품질관리 - 공정불량실적

• 검색 및 조회

[전체 화면]

새로고침

제조LOT번호

공정번호

불량유형

검사일자

불량유형1

불량유형2

연도-월-일

연도-월-일

☰ 공정별 불량실적목록

구분		NO	제품명	단위	총생산수량	양품수량	불량수량	불량률(%)		
제조LOT번호	공정번호	불량유형1	불량유형2							
300,001	1,001	손상	파손	1	의자-A	ea	4,388	4,360	12	0.270
300,001	1,001	형태변형	꺼짐(찌그러짐)	2	의자-A	ea	2,500	2,480	3	0.120
300,001	1,001	형태변형	휠	3	의자-A	ea	4,388	4,360	13	0.300
300,001	1,011	도장불량	색상 불균일	4	의자-A	ea	600	595	3	0.500
300,001	1,011	형태변형	꺼짐(찌그러짐)	5	의자-A	ea	600	595	2	0.330

동일한 [LOT번호], [공정번호]의
다른 [불량유형]의 누적된 데이터를 검색 및 조회

재고관리 - 제품재고조회

• 검색 및 조회

제품재고조회

제품번호: 제품명: 제품용도:

검색 키워드: 검색

NO	제품번호	제품용도	제품규격	제품명	단위	재고수량
1	10,000,001	완제품	성인용	의자-A	EA	595
2	10,000,006	완제품	싱글	침대-A	EA	600
3	10,000,011	완제품	1인용	소파-A	EA	57
4	10,000,016	완제품	1인용	책상-A	EA	78
5	10,000,017	완제품	1인용	책상-B	EA	642
6	10,000,024	완제품	5단	서랍장-D	EA	1,320
7	10,000,097	반제품	5T	소나무 다리	EA	2,503
8	10,000,098	반제품	10T	소나무 판	EA	3,500
9	10,000,099	반제품	5T	소나무 프레임1	EA	2,305
10	10,000,100	반제품	5T	소나무 프레임2	EA	230
11	10,000,101	반제품	5T	소나무 프레임3	EA	120

검색 키워드
datalist로 조회하면서 입력 가능

제품번호: 제품재고목록

NO	제품번호
1	10000001
2	10000006
3	10000011
4	10000016
5	10000017
6	10000024
7	10000097
8	10000098
9	10000099
10	10000100
11	10000101
12	10000102
13	10000103
14	10000104
15	10000105

느낀 점(정수하)

[좋았던 점]

- 멋진 리더와 팀원을 만나서, 소통이 잘 됨.
- 설계, 서류 정리 등 탄탄하게 한 부분이 좋았음.
- 배운 지식을 활용하는 좋은 기회라고 생각함.
- 깃, 깃허브의 이해도를 높일 수 있었음.
- 팀원이 많은 만큼, 다양한 코드를 볼 수 있어서 좋음.

[아쉬웠던 점]

- 개발 기간이 짧아서 아쉬움.
- 쿼리문이 어려워서 시간이 좀 걸린 것이 아쉬움.

자체 평가 의견

[좋았던 점]

- Ⓐ 깃과 깃허브의 활용도가 높아 진행도를 쉽게 알 수 있던 점
- Ⓐ 서류와 설계에 많은시간을 투자한만큼
개발에 대한 각자의 이해도가 높았던 점
- Ⓐ 소통과 연락이 잘 되어
자유롭게 의견을 표현하고 피드백이 빨랐던 점
- Ⓐ 개발 과정에서 팀원들이 작성한 코드를
서로 검토하면서 다양한 프로그래밍 스타일과
해결 방안을 보고 배울 수 있던 점

[아쉬웠던 점]

- Ⓐ 긴 설계기간(13일), 짧은 개발기간(11일)

[개선 사항 및 활용 계획]

- Ⓐ 각자의 페이지 중 중복 데이터를 완벽하게 막지 못해
겹치는 데이터가 생성되어 오류 발생함.
그 부분에 있어 리팩토링 할 예정
- Ⓐ 개발하고자 한 로직과 기능은 다 구현했지만
오픈 API를 이용해 다양한 기능을 넣을 예정

감사합니다

THANK YOU



이메일

배지현: mkiopl01@gmail.com

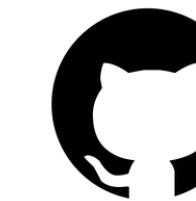
김다은: gofl779@gmail.com

이시연: sieoh@naver.com

이현주: lunitaropez@gmail.com

조다혜: dahyejo461@gmail.com

정수하: jsh0210hey@gmail.com



깃허브

배지현: github.com/jihyeon00

김다은: github.com/dan3319

이시연: github.com/sieoh

이현주: github.com/icanbewhatever

조다혜: github.com/ChoDaHye

정수하: github.com/heyJSH

