

QMS Portfolio

이름	이재혁
Mobile	010-2957-8599
E-mail	dlwogur8599@gmail.com
Address	경기도 의정부시

개발 환경

Backend

- Java
- Spring Boot
- MyBatis

Frontend

- JSP
- JQuery
- Bootstrap

Database

- Oracle DB

3인 팀 프로젝트 - QMS

소개

품질관리시스템에서 제품과 자재를 관리하는 전반적인 영역을 문서화하고 기록/ 관리할 수 있는 페이지

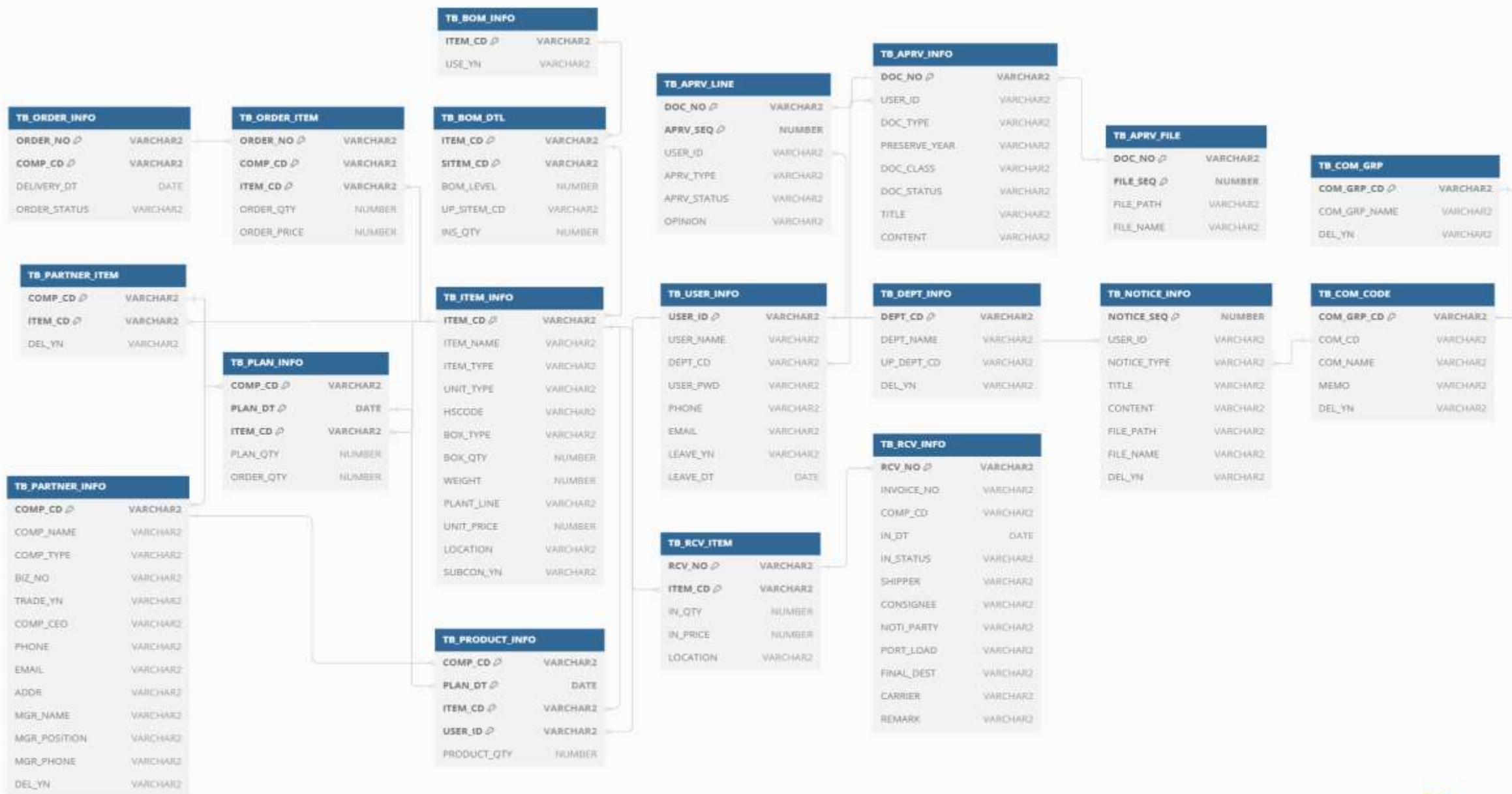
소요기간

24/06/15 ~ 24/07/10

담당한 주요 기능

- 엑셀 업로드
- 엑셀 다운
- 다국어 처리

DATABASE ERD



담당 주요 기능 설명

엑셀 업로드 기능 설명

엑셀 업로드 버튼을 누르면 모달창이 켜진다.

홈 / 자재관리 / 제품조회

품번

품명

제품유형

===선택===▼

조회

신규

엑셀

엑셀업로드

No	품목코드	품목명	단위	생산라인	BOX규격	재고위치	등록일
----	------	-----	----	------	-------	------	-----

© Copyright EZEN ACADEMY. All Rights Reserved

품목 엑셀 업로드

엑셀파일

파일 선택

선택된 파일 없음

엑셀업로드닫기

엑셀 업로드 기능 설명

Input type을 file로 설정하고 accept=".xlsx, .xls" 를 적어서 엑셀 형식의 파일만 올릴 수 있게 설정한다.

```
233      ← 엑셀 모달 시작 →
234      <div class="modal fade" id="excelModal" tabindex="-1">
235      <div class="modal-dialog modal-xl modal-dialog-centered">
236      <div class="modal-content">
237      <div class="modal-header">
238          <h5 class="modal-title" style="text-align:center; width:100%">품목 엑셀 업로드</h5>
239          <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>
240      </div>
241      <div class="modal-body">
242      <form id="excelform" method="POST" enctype="multipart/form-data">
243      <table id="modaltable" style="width:100%; height:100%;">
244      <tr>
245          <th>엑셀파일</th>
246          <td><input type="file" id="excelFile" name="excelFile" accept=".xlsx, .xls"></td>
247      </tr>
248      </table>
249      </form>
250      </div>
251      <div class="modal-footer">
252          <button type="button" class="btn btn-primary" onclick = 'upload();'>엑셀업로드</button>
253          <button type="button" class="btn btn-secondary" data-bs-dismiss="modal">닫기</button>
254      </div>
255      </div>
256      </div>
257      </div>
258      ← 엑셀 모달 끝 →
```

엑셀 업로드 기능 설명

파일을 선택하고 업로드 버튼을 누르면 아래의 스크립트를 따라 엑셀 파일인지 확인한다.

```
// 엑셀 업로드
function excelUpload(){
    $('#excelform')[0].reset();
    $('#excelModal').modal('show');
}
function upload(){
    var file = $('#excelFile').val();

    if(file == "" || file == null){
        alert("파일을 선택해주세요.");
        return false;
    }else if(!checkFileType(file)){
        alert("엑셀 파일만 업로드 가능합니다.");
        return false;
    }

    if(confirm("업로드 하시겠습니까")){
        excelCall_server(excelform, "/item/excelUpload", uploadExcel);
    }
}

// 엑셀 파일인지 확인
function checkFileType(filePath){
    var fileFormat = filePath.split(".");
    if(fileFormat.indexOf("xls") > -1 || fileFormat.indexOf("xlsx") > -1){
        return true;
    }else{
        return false;
    }
}
```


엑셀 업로드 기능 설명

엑셀 파일로 확인되면 호출되는 컨트롤러

XSSFWorkbook, XSSFSheet, XSSFRow를 사용해서
엑셀 데이터를 문자열로 가져온다.

```
// 엑셀 업로드
@RequestMapping("/item/excelUpload")
@ResponseBody
public int excelUpload(@ModelAttribute ExcelVO vo, HttpServletRequest request) throws Exception {
    HttpSession session = request.getSession();
    UserInfoVO userVo = (UserInfoVO) session.getAttribute("MallUser");
    vo.setRegUserId(userVo.getUserId());

    // 엑셀 파일을 읽고 첫 번째 시트를 가져온다.
    XSSFWorkbook workbook = new XSSFWorkbook(vo.getExcelFile().getInputStream());
    XSSFSheet worksheet = workbook.getSheetAt(0);

    // 셀 데이터를 문자열로 변환
    DataFormatter formatter = new DataFormatter();
    int cnt = 0;

    // 행만큼의 길이 설정하고 셀 데이터 넣기 위한 리스트 선언
    for (int i = 1; i < worksheet.getPhysicalNumberOfRows(); i++) { // 헤더 부분 다음 행을 가져오기 위해 i=1로 선언
        XSSFRow row = worksheet.getRow(i);
        List<String> cellDataList = new ArrayList<>();

        // 조회하는 첫 번째 열이 비어있는 경우 반복문 종료
        if(formatter.formatCellValue(row.getCell(1)) == "") {
            break;
        }

        // 각 열의 셀 데이터만큼 길이 설정, 위에 선언한 리스트에 각 셀의 데이터 삽입
        for (int j = 1; j < row.getPhysicalNumberOfCells(); j++) { // NO 부분 다음 열을 가져오기 위해 j=1로 선언
            String cellValue = formatter.formatCellValue(row.getCell(j));
            vo.setItemCd(formatter.formatCellValue(row.getCell(1)));
            cellDataList.add(cellValue);
        }

        // 세션에 접속된 아이디를 리스트에 추가
        cellDataList.add(vo.getRegUserId());
        // 기존에 있던 데이터면 지우고 각 리스트 db에 저장
        service.deleteExcel(vo);
        cnt += service.insertExcel(cellDataList);
    }

    workbook.close();

    return cnt;
}
```

엑셀 업로드 기능 설명

insertExcel 쿼리

```
<insert id="insertExcel" parameterType="java.util.List">
    INSERT INTO TB_ITEM_INFO(ITEM_CD, ITEM_NAME, ITEM_TYPE, UNIT_TYPE, HSCODE, BOX_TYPE, BOX_SIZE)
    VALUES (
        <foreach collection="cellDataList" item="item" separator=",">
            #{item}
        </foreach>
        ,SYSDATE
    )
</insert>
```

쿼리가 정상적으로 작동하면 실행된 쿼리의 횟수만큼 아래의 스크립트 출력

```
function uploadExcel(cnt){
    if(cnt>0){
        alert("총 "+cnt+"개의 데이터를 입력했습니다");
        $('#excelModal').modal('hide');
        search();
    }else{
        alert("업로드실패");
    }
}
```

엑셀 업로드 기능 설명

localhost:8080 내용:

총 2개의 데이터를 입력했습니다

확인

품명

제품유형

===선택=== ▼

조회

품목명

단위

생산라인

BOX규격

재

© Copyright EZEN ACADEMY. All Rights Reserved

품목 엑셀 업로드

×

엑셀파일 파일 선택 품목정보.xlsx

엑셀업로드

닫기

엑셀 다운로드 기능 설명

생산계획조회 페이지

거래처들의 생산계획을 조회하고 엑셀로 다운로드 받을 수 있다.

[홈](#) / [생산관리](#) / [생산계획조회](#)

계 획 년 월

2024

64

五

거래처

주식회사 블루



조호

엑셀 다운로드

[illegible]

엑셀 다운로드 기능 설명

엑셀 다운로드 버튼을 누르면 /plan/planQtyExcel 이라는 요청을 받아 우측의 컨트롤러로 연결된다.

Calender를 사용해서 조회한 달의 일수를 계산한다.

```
// 계획수량 다운로드
@PostMapping("/plan/planQtyExcel")
public ResponseEntity<byte[]> downloadPlanExcel(@ModelAttribute("PlanInfoV0")PlanInfoV0 vo) throws Exception {
    Map<String, Object> parameters = new HashMap<>();

    Calendar cal = Calendar.getInstance();

    cal.set(Integer.parseInt(vo.getPlanYear()), Integer.parseInt(vo.getPlanMonth())-1, 1);

    System.out.println(cal.getActualMaximum(Calendar.DAY_OF_MONTH));

    String[] headers = Constant.PLANQTY_HEADER;
    String[] dateHeaders = new String[cal.getActualMaximum(Calendar.DAY_OF_MONTH)];
    String[] dateCol = new String[cal.getActualMaximum(Calendar.DAY_OF_MONTH)];

    List<Map<String, Object>> dataList = excelService.selectPlanQtyToExcel(vo);
    for(int i = 1; i ≤ cal.getActualMaximum(Calendar.DAY_OF_MONTH); i++) {
        dateHeaders[i-1] = i+"일";
        dateCol[i-1] = "d"+i;
    }

    String[] arr3 = new String[headers.length + dateHeaders.length];
    System.arraycopy(headers, 0, arr3, 0, headers.length);
    System.arraycopy(dateHeaders, 0, arr3, headers.length, dateHeaders.length);

    //String[] columns = ExcelConstant.PLAN_COLUMN;
    String[] columns = Constant.PLANQTY_COLUMN;

    String[] arrCol = new String[columns.length + dateCol.length];
    System.arraycopy(columns, 0, arrCol, 0, columns.length);
    System.arraycopy(dateCol, 0, arrCol, columns.length, dateCol.length);

    String sheetName = "생산계획수량";
    String fileName = "PlanQty.xlsx";
    return excelService.createExcelFile(dataList, arrCol, arr3, fileName, sheetName);
}
```

엑셀 다운로드 기능 설명

```
String[] headers = Constant.PLANQTY_HEADER;  
String[] dateHeaders = new String[cal.getActualMaximum(Calendar.DAY_OF_MONTH)];  
String[] dateCol = new String[cal.getActualMaximum(Calendar.DAY_OF_MONTH)];
```

String[] headers 선언을 위해 먼저 Constant 클래스를 생성해서 지정해준다.

```
package com.qms.Util;  
  
public class Constant {  
  
    // 계획수량/생산실적 엑셀  
    public static final String[] PLANQTY_HEADER = {"거래처명", "품번", "품명", "SUM"};  
  
    public static final String[] PLANQTY_COLUMN = {"compName", "itemCd", "itemName", "SUM"};  
}
```

Headers에 정상적으로 값이 들어옴

▼ headers	String[4] (id=208)
> ▲ [0]	"거래처명" (id=210)
> ▲ [1]	"품번" (id=214)
> ▲ [2]	"품명" (id=215)
> ▲ [3]	"SUM" (id=216)

엑셀 다운로드 기능 설명

```
String[] headers = Constant.PLANQTY_HEADER;  
String[] dateHeaders = new String[cal.getActualMaximum(Calendar.DAY_OF_MONTH)];  
String[] dateCol = new String[cal.getActualMaximum(Calendar.DAY_OF_MONTH)];
```

다음으로 dateHeades 와 dateCol은 Calender의 내장메소드를 사용해서 지정해준다.

voPlaninfoVO (id=136)

parametersHashMap<K,V> (id=139)

calGregorianCalendar (id=183)

headersString[4] (id=208)

[0]"거래처명" (id=210)

[1]"품번" (id=214)

[2]"품명" (id=215)

[3]"SUM" (id=216)

dateHeadersString[30] (id=218)

dateColString[30] (id=225)

dataListArrayList<E> (id=230)

arr3String[34] (id=385)

columnsString[4] (id=390)

<Choose a previously entered expression>

[1월, 2월, 3월, 4월, 5월, 6월, 7월, 8월, 9월, 10월, 11월, 12월, 13월, 14월,

headersString[4] (id=208)

[0]"거래처명" (id=210)

[1]"품번" (id=214)

[2]"품명" (id=215)

[3]"SUM" (id=216)

dateHeadersString[30] (id=218)

dateColString[30] (id=225)

dataListArrayList<E> (id=230)

arr3String[34] (id=385)

columnsString[4] (id=390)

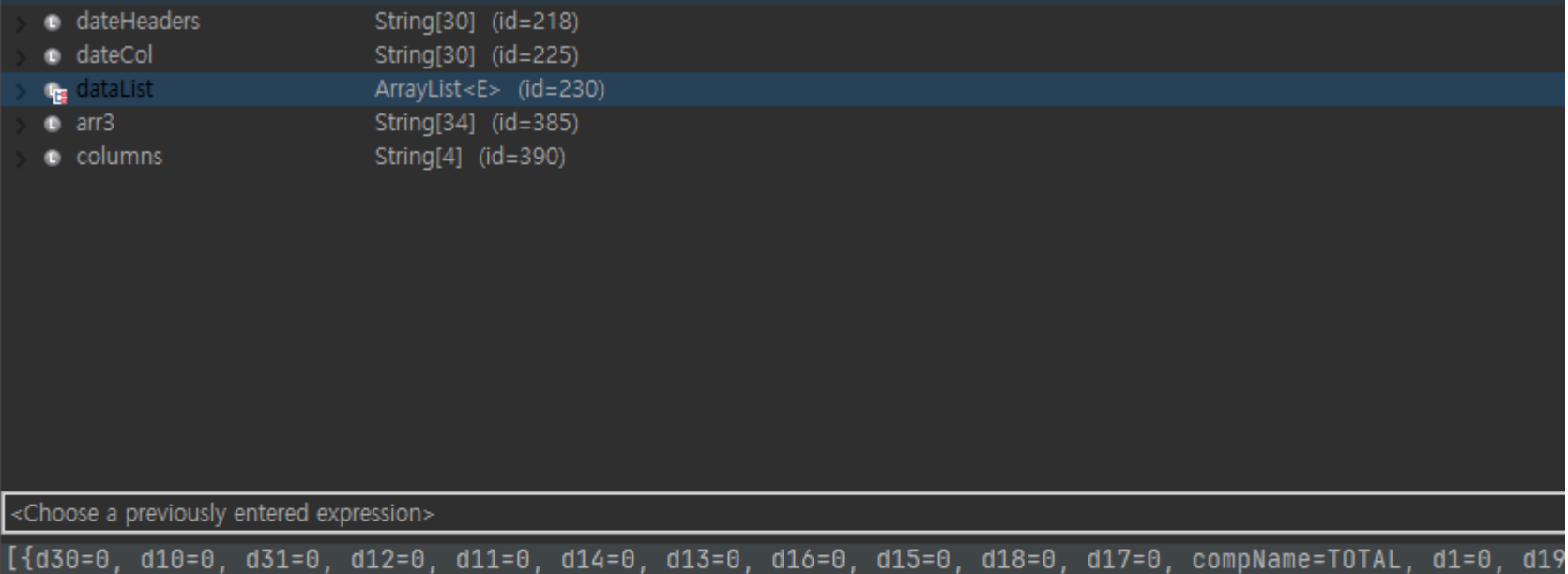
<Choose a previously entered expression>

[d1, d2, d3, d4, d5, d6, d7, d8, d9, d10, d11, d12, d13, d14, d

엑셀 다운로드 기능 설명

```
List<Map<String, Object>> dataList = excelService.selectPlanQtyToExcel(vo);
for(int i = 1; i ≤ cal.getActualMaximum(Calendar.DAY_OF_MONTH); i++) {
    dateHeaders[i-1] = i+"월";
    dateCol[i-1] = "d"+i;
}
```

excelService의 selectPlanQtyToExcel을 호출하여 dataList의 값을 넣어준다.



>	dateHeaders	String[30] (id=218)
>	dateCol	String[30] (id=225)
>	dataList	ArrayList<E> (id=230)
>	arr3	String[34] (id=385)
>	columns	String[4] (id=390)

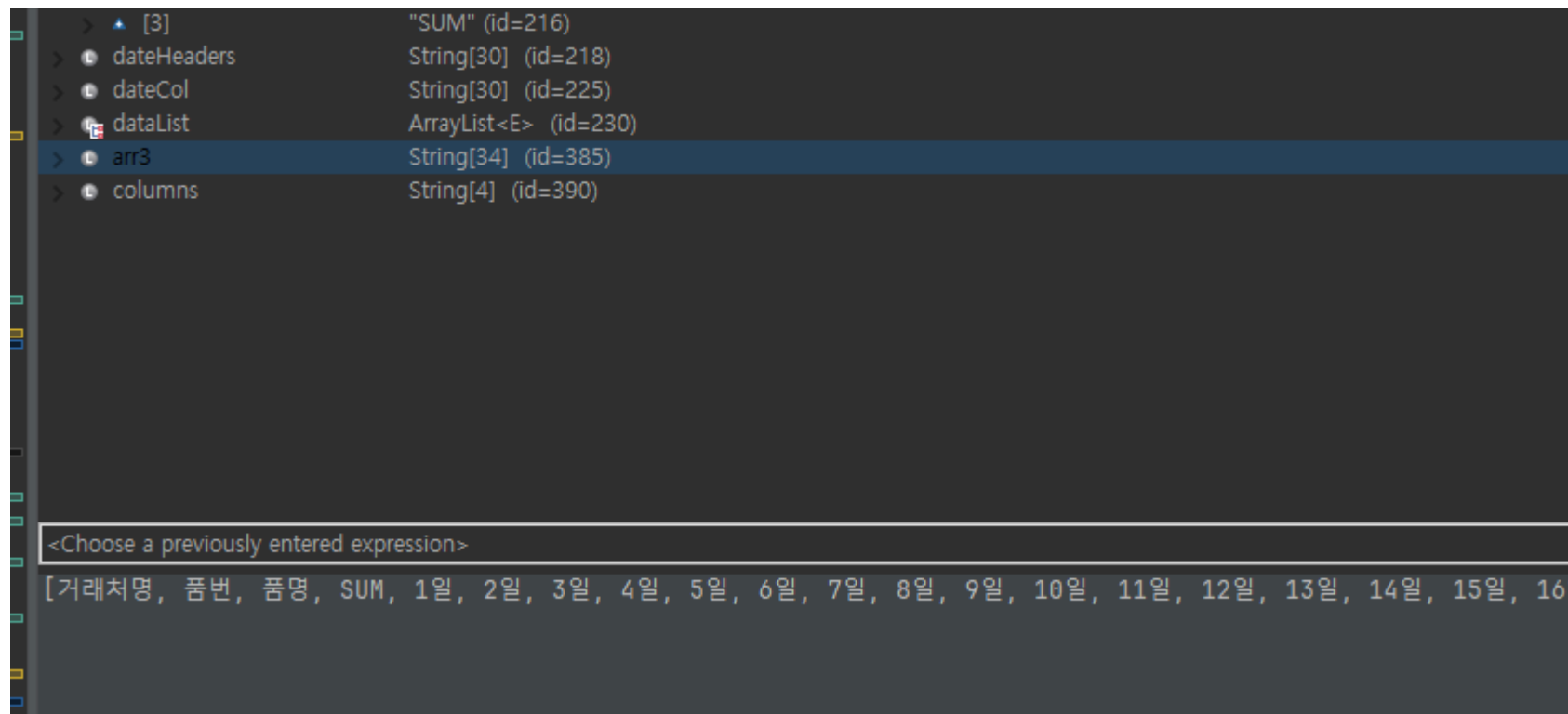
<Choose a previously entered expression>

```
[{d30=0, d10=0, d31=0, d12=0, d11=0, d14=0, d13=0, d16=0, d15=0, d18=0, d17=0, compName=TOTAL, d1=0, d19=0, ...}]
```


엑셀 다운로드 기능 설명

```
String[] arr3 = new String[headers.length + dateHeaders.length];  
System.arraycopy(headers, 0, arr3, 0, headers.length);  
System.arraycopy(dateHeaders, 0, arr3, headers.length, dateHeaders.length);
```

Arr3라는 배열을 선언하고 arraycopy를 사용하여 값을 넣어준다.



엑셀 다운로드 기능 설명

모든 값의 설정이 끝났으면 마지막으로
createExcelFile 메소드로 엑셀 생성

```
public ResponseEntity<byte[]> createExcelFile(List<Map<String, Object>> dataList, String[] columns, String[] headers,
try {
    Workbook workbook = new XSSFWorkbook();
    Sheet sheet = workbook.createSheet(sheetName); // 시트 이름 설정

    // Header
    Font headerFont = workbook.createFont();
    headerFont.setBold(true);
    CellStyle headerCellStyle = workbook.createCellStyle();
    headerCellStyle.setFont(headerFont);
    Row headerRow = sheet.createRow(0);
    for (int i = 0; i < headers.length; i++) {
        Cell cell = headerRow.createCell(i);
        cell.setCellValue(headers[i]);
        cell.setCellStyle(headerCellStyle);
    }

    // Data
    int rowNum = 1;
    for (Map<String, Object> data : dataList) {
        Row row = sheet.createRow(rowNum++);
        int cellNum = 0;
        for (String column : columns) {
            Object value = data.get(column);
            row.createCell(cellNum++).setCellValue(value != null ? value.toString() : "");
        }
    }

    ByteArrayOutputStream out = new ByteArrayOutputStream();
    workbook.write(out);
    workbook.close();

    byte[] excelBytes = out.toByteArray();

    HttpHeaders responseHeaders = new HttpHeaders();
    responseHeaders.setContentType(MediaType.APPLICATION_OCTET_STREAM);
    responseHeaders.setContentDispositionFormData("attachment", fileName);
    responseHeaders.setContentLength(excelBytes.length);

    return ResponseEntity.ok()
        .headers(responseHeaders)
        .body(excelBytes);
}
catch (IOException e) {
    e.printStackTrace();
    return null; // Error handling
}
```

엑셀 다운로드 기능 설명

생성된 엑셀파일 모습

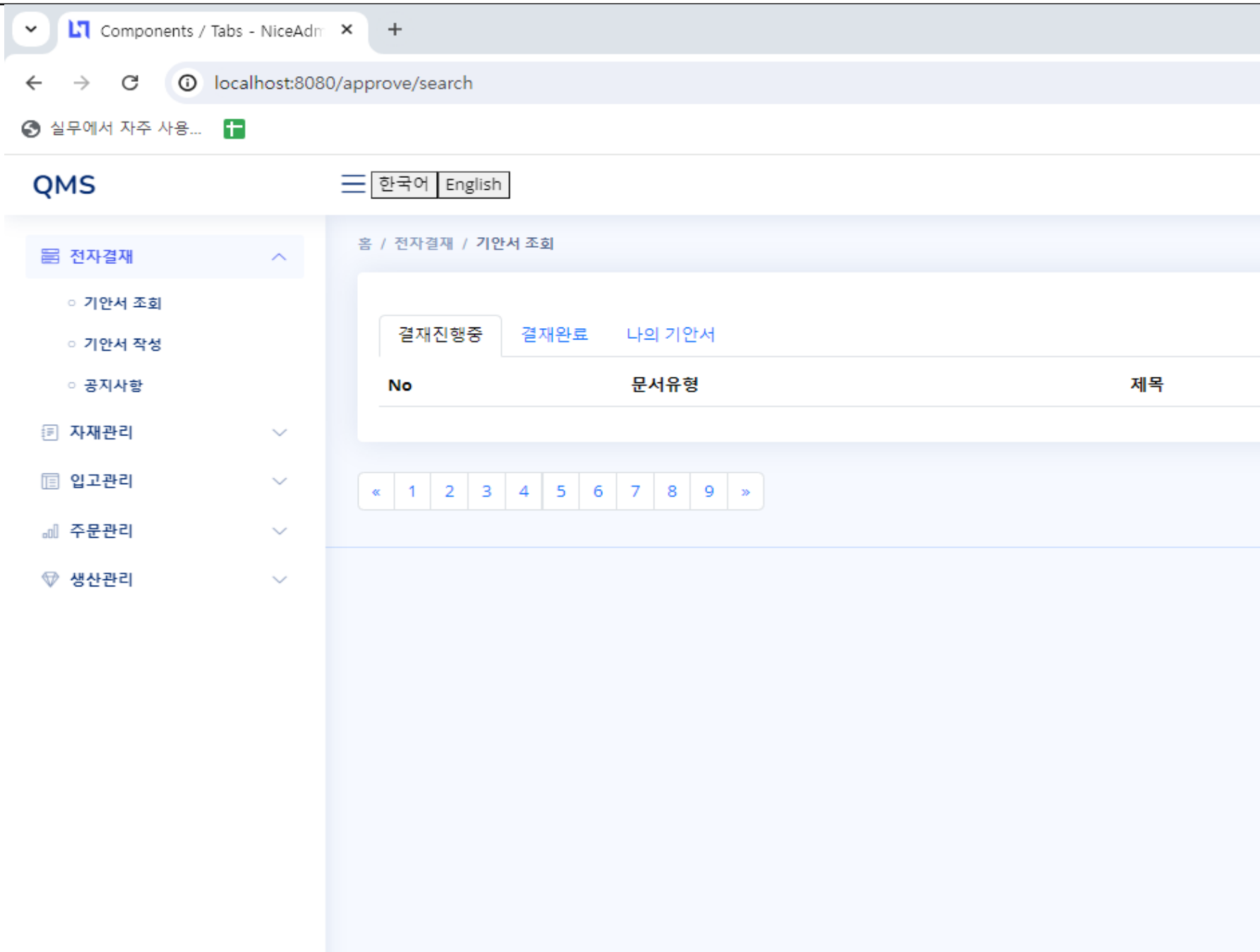
[illegible]

다국어 처리 기능 설명

기본 페이지 화면

Url이 localhost:8080/approve/search 로 표시됩니다.

모든 페이지에 적용한 헤더 부분에
다국어 처리를 할 수 있는 버튼(한국어, English)를 만들었습니다.



다국어 처리 기능 설명

영어 버튼을 클릭 시 url 주소가
localhost:8080/approve/search?lang=en
으로 변경되고 헤더와 사이드바 메인 컨테
츠 영역까지 전부 영어로 변경됩니다.

The screenshot shows a web browser window with the address bar displaying `localhost:8080/approve/search?lang=en`. The browser tab is titled "Components / Tabs - NiceAdmin". The application header features the "QMS" logo and a language switcher with buttons for "한국어" (Korean) and "English". The "English" button is currently selected. The left sidebar contains a menu with the following items: "Electric-Aprv" (expanded), "Aprv-Search", "Aprv-Write", "Notice", "Product-Management", "Receiving-Management", "Order-Management", and "Production-Management". The main content area shows a breadcrumb trail: "Home / Electric-Aprv / Aprv-Search". Below the breadcrumb, there are three tabs: "Approving", "Approved", and "My Approve". The "Approved" tab is active. A table is displayed with the following headers: "No", "DocumentType", and "Title". Below the table, there is a pagination control showing a range of numbers from 1 to 9, with "«" and "»" for navigation.

다국어 처리 기능 설명

한국어 버튼을 다시 클릭 시 url 주소 뒷 부분이 ?lang=ko로 변경되고 페이지가 다시 한국어로 표시됩니다.

The screenshot shows a web browser window with the address bar displaying `localhost:8080/approve/search?lang=ko`. The browser's address bar also shows a bookmark labeled "실무에서 자주 사용...". The web application is titled "QMS" and has a language switcher in the top right corner with buttons for "한국어" (Korean) and "English". The "한국어" button is currently selected. The left sidebar contains a menu with the following items: "전자결재" (Electronic Approval), "자재관리" (Material Management), "입고관리" (Inventory Management), "주문관리" (Order Management), and "생산관리" (Production Management). The "전자결재" menu is expanded, showing sub-items: "기안서 조회" (Proposal Search), "기안서 작성" (Proposal Creation), and "공지사항" (Notice). The main content area displays a breadcrumb trail: "홈 / 전자결재 / 기안서 조회". Below the breadcrumb, there are three tabs: "결재진행중" (Approval in Progress), "결재완료" (Approval Completed), and "나의 기안서" (My Proposals). The "결재진행중" tab is active. Below the tabs, there is a table with the following columns: "No", "문서유형" (Document Type), and "제목" (Title). The table is currently empty. At the bottom of the table, there is a pagination control showing the current page is 1 out of 9 pages, with buttons for "«", "1", "2", "3", "4", "5", "6", "7", "8", "9", and "»".

다국어 처리 기능 설명

다국어 처리를 해주기 위해 제일 먼저 만들어준 LocaleConfig 파일입니다.

스프링부트가 제공하는 LocaleResolver 인터페이스를 사용했습니다.

Locale 정보를 session에서 가져오게 SessionLocaleResolver로 설정했습니다.

LocaleChangeInterceptor를 선언해서 lang=en 과 같은 요청이 들어오면 Locale을 en으로 변경해주게 만들었습니다.

```
LocaleConfig.java X menu.jsp
1 package com.qms.config;
2
3 import org.springframework.context.annotation.Bean;
12
13 @Configuration
14 public class LocaleConfig implements WebMvcConfigurer{
15
16     @Bean
17     public LocaleResolver localeResolver() {
18         SessionLocaleResolver sessionLocaleResolver = new SessionLocaleResolver();
19         sessionLocaleResolver.setDefaultLocale(Locale.KOREAN); // 기본 언어 설정
20         return sessionLocaleResolver;
21     }
22
23     @Bean
24     public LocaleChangeInterceptor localeChangeInterceptor() {
25         LocaleChangeInterceptor localeChangeInterceptor = new LocaleChangeInterceptor();
26         localeChangeInterceptor.setParamName("lang"); // 언어 변경을 위한 파라미터 이름
27         return localeChangeInterceptor;
28     }
29
30     @Override
31     public void addInterceptors(InterceptorRegistry registry) {
32         registry.addInterceptor(localeChangeInterceptor());
33     }
34 }
35
```

다국어 처리 기능 설명

언어 변경을 용이하게 하기 위해서 단순 클릭으로 언어를 변경할 수 있게 버튼을 만들고 스크립트를 설정해줬습니다.

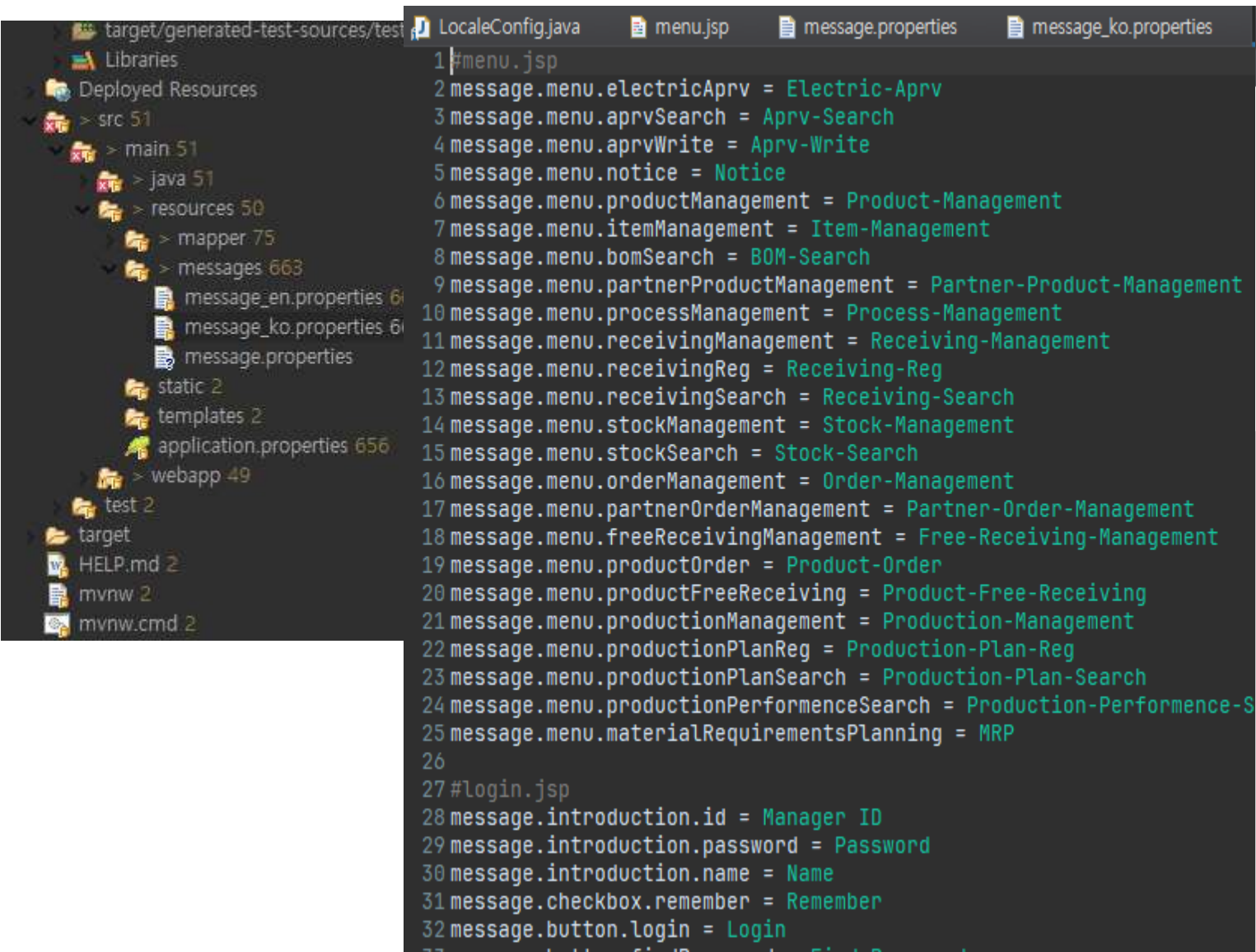
현재 주소 url을 가져와 인덱스 번호를 사용하여 설정된 언어가 있는 지 없는 지 확인 후 없으면 langParam을 넣어서 페이지를 변경하고, 현재 설정된 언어가 있다면 '?' 기준으로 뒷부분을 지우고 다시 langParam를 넣어주는 방법을 사용했습니다.

```
16
17● <header id="header" class="header fixed-top d-flex align-items-center">
18
19● <div class="d-flex align-items-center justify-content-between">
20● <a href="index.html" class="logo d-flex align-items-center">
21 
22 <span class="d-none d-lg-block">QMS</span>
23 </a>
24 <i class="bi bi-list toggle-sidebar-btn"></i>
25 <button type="button" onclick="Korean();">한국어</button>
26 <button type="button" onclick="English();">English</button>
27 </div><!-- End Logo -->
28
29
```

```
LocaleConfig.java  menu.jsp X
234 // 언어 설정
235 function Korean() {
236     var currentUrl = window.location.href;
237     var langParam = 'lang=ko';
238
239     // 이미 설정된 언어가 있는지 확인
240     if (currentUrl.indexOf('?') !== -1) {
241         // 있는 경우
242         currentUrl = currentUrl.substring(0, currentUrl.indexOf('?'));
243         currentUrl += '?' + langParam;
244     } else {
245         // 없는 경우
246         currentUrl += '?' + langParam;
247     }
248
249     window.location.href = currentUrl;
250 }
251 function English() {
252     var currentUrl = window.location.href;
253     var langParam = 'lang=en';
254
255     // 이미 설정된 언어가 있는지 확인
256     if (currentUrl.indexOf('?') !== -1) {
257         // 있는 경우
258         currentUrl = currentUrl.substring(0, currentUrl.indexOf('?'));
259         currentUrl += '?' + langParam;
260     } else {
261         // 없는 경우
262         currentUrl += '?' + langParam;
263     }
264
265     window.location.href = currentUrl;
266 }
```

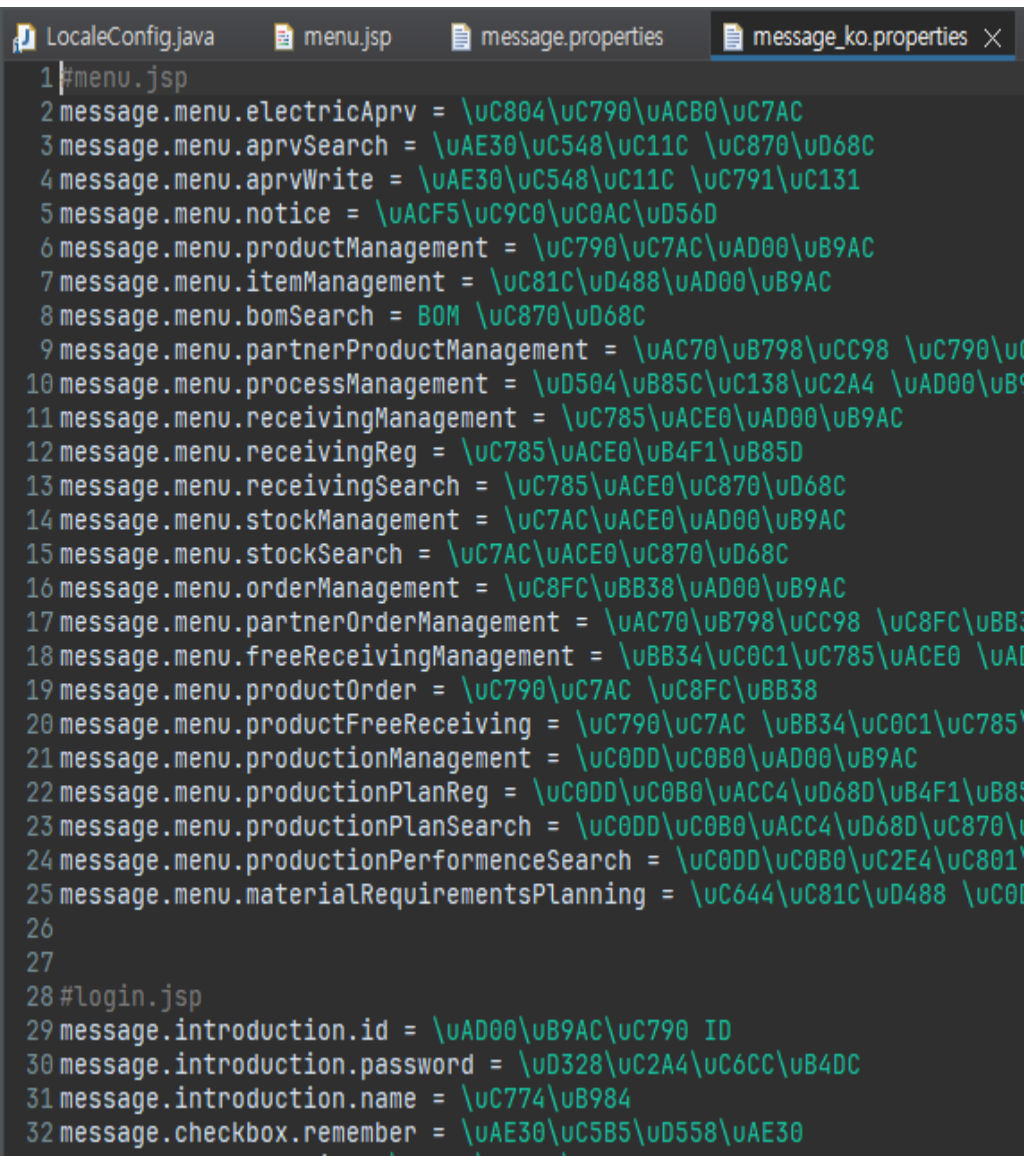

다국어 처리 기능 설명

다음으로 resources 폴더 내부에 messages 폴더를 만들어주고 default 언어 파일과 한국어, 영어 파일을 만들어 줬습니다.



```
target/generated-test-sources/test-annotations
Libraries
Deployed Resources
src 51
  main 51
    java 51
      resources 50
        mapper 75
        messages 663
          message_en.properties 61
          message_ko.properties 61
          message.properties
        static 2
        templates 2
        application.properties 656
      webapp 49
    test 2
  target
  HELP.md 2
  mvnw 2
  mvnw.cmd 2
```

```
LocaleConfig.java menu.jsp message.properties message_ko.properties
1 #menu.jsp
2 message.menu.electricAprv = Electric-Aprv
3 message.menu.aprvSearch = Aprv-Search
4 message.menu.aprvWrite = Aprv-Write
5 message.menu.notice = Notice
6 message.menu.productManagement = Product-Management
7 message.menu.itemManagement = Item-Management
8 message.menu.bomSearch = BOM-Search
9 message.menu.partnerProductManagement = Partner-Product-Management
10 message.menu.processManagement = Process-Management
11 message.menu.receivingManagement = Receiving-Management
12 message.menu.receivingReg = Receiving-Reg
13 message.menu.receivingSearch = Receiving-Search
14 message.menu.stockManagement = Stock-Management
15 message.menu.stockSearch = Stock-Search
16 message.menu.orderManagement = Order-Management
17 message.menu.partnerOrderManagement = Partner-Order-Management
18 message.menu.freeReceivingManagement = Free-Receiving-Management
19 message.menu.productOrder = Product-Order
20 message.menu.productFreeReceiving = Product-Free-Receiving
21 message.menu.productionManagement = Production-Management
22 message.menu.productionPlanReg = Production-Plan-Reg
23 message.menu.productionPlanSearch = Production-Plan-Search
24 message.menu.productionPerformanceSearch = Production-Performance-S
25 message.menu.materialRequirementsPlanning = MRP
26
27 #login.jsp
28 message.introduction.id = Manager ID
29 message.introduction.password = Password
30 message.introduction.name = Name
31 message.checkbox.remember = Remember
32 message.button.login = Login
33
```



```
LocaleConfig.java menu.jsp message.properties message_ko.properties
1 #menu.jsp
2 message.menu.electricAprv = \uC804\uC790\uCACB\uC7AC
3 message.menu.aprvSearch = \uAE30\uC548\uC11C \uC870\uD68C
4 message.menu.aprvWrite = \uAE30\uC548\uC11C \uC791\uC131
5 message.menu.notice = \uACF5\uC9C0\uC0AC\uD56D
6 message.menu.productManagement = \uC790\uC7AC\uAD00\uB9AC
7 message.menu.itemManagement = \uC81C\uD488\uAD00\uB9AC
8 message.menu.bomSearch = BOM \uC870\uD68C
9 message.menu.partnerProductManagement = \uAC70\uB798\uCC98 \uC790\u
10 message.menu.processManagement = \uD504\uB85C\uC138\uC2A4 \uAD00\uB
11 message.menu.receivingManagement = \uC785\uACE0\uAD00\uB9AC
12 message.menu.receivingReg = \uC785\uACE0\uB4F1\uB85D
13 message.menu.receivingSearch = \uC785\uACE0\uC870\uD68C
14 message.menu.stockManagement = \uC7AC\uACE0\uAD00\uB9AC
15 message.menu.stockSearch = \uC7AC\uACE0\uC870\uD68C
16 message.menu.orderManagement = \uC8FC\uBB38\uAD00\uB9AC
17 message.menu.partnerOrderManagement = \uAC70\uB798\uCC98 \uC8FC\uBB
18 message.menu.freeReceivingManagement = \uBB34\uC0C1\uC785\uACE0 \uA
19 message.menu.productOrder = \uC790\uC7AC \uC8FC\uBB38
20 message.menu.productFreeReceiving = \uC790\uC7AC \uBB34\uC0C1\uC785
21 message.menu.productionManagement = \uC0DD\uC0B0\uAD00\uB9AC
22 message.menu.productionPlanReg = \uC0DD\uC0B0\uACC4\uD68D\uB4F1\uB8
23 message.menu.productionPlanSearch = \uC0DD\uC0B0\uACC4\uD68D\uC870\u
24 message.menu.productionPerformanceSearch = \uC0DD\uC0B0\uC2E4\uC801
25 message.menu.materialRequirementsPlanning = \uC644\uC81C\uD488 \uC0
26
27
28 #login.jsp
29 message.introduction.id = \uAD00\uB9AC\uC790 ID
30 message.introduction.password = \uD328\uC2A4\uC6CC\uB4DC
31 message.introduction.name = \uC774\uB984
32 message.checkbox.remember = \uAE30\uC5B5\uD558\uAE30
```

다국어 처리 기능 설명

messages 파일에서 선언해준 언어를 실제로 js
p에서 사용하기 위해 tag 라이브러리를 선언해
주고 텍스트가 위치할 부분에
<spring:message code = ""/> 를 넣어
마무리 했습니다.

```
<!-- ===== Sidebar ===== -->
<%@ taglib prefix="spring" uri="http://www.springframework.org/tags" %>
<aside id="sidebar" class="sidebar">

    <ul class="sidebar-nav" id="sidebar-nav">

        <li class="nav-item">
            <a class="nav-link collapsed" data-bs-target="#components-nav" data-bs-toggle="collapse" href="#">
                <i class="bi bi-menu-button-wide"></i><span><spring:message code="message.menu.electricAprv"/></span>
            </a>
            <ul id="components-nav" class="nav-content collapse " data-bs-parent="#sidebar-nav">
                <li>
                    <a href="/approve/search">
                        <i class="bi bi-circle"></i><span><spring:message code="message.menu.aprvSearch"/></span>
                    </a>
                </li>
                <li>
                    <a href="/approve/write">
                        <i class="bi bi-circle"></i><span><spring:message code="message.menu.aprvWrite"/></span>
                    </a>
                </li>
                <li>
                    <a href="/board/list">
                        <i class="bi bi-circle"></i><span><spring:message code="message.menu.notice"/></span>
                    </a>
                </li>
            </ul>
        </li>
    </ul>
</aside>
```