

## tiles 적용




tiles를 통해 레이아웃을 적용해  
서 모든 페이지에서 공통적으로  
사용했습니다.


## tiles 적용

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <!DOCTYPE tiles-definitions PUBLIC
3 "-//Apache Software Foundation//DTD Tiles Configuration 2.0//EN"
4 "http://tiles.apache.org/dtds/tiles-config_2_0.dtd">
5 <tiles-definitions>
6   <definition name="baseLayout" template="/WEB-INF/views/tiles/Layout.jsp">
7     <put-attribute name="title" value="레이아웃" />
8     <put-attribute name="header" value="/WEB-INF/views/tiles/header.jsp" />
9     <put-attribute name="nav" value="/WEB-INF/views/tiles/nav.jsp" />
10    <put-attribute name="content" value="" />
11    <put-attribute name="footer" value="/WEB-INF/views/tiles/footer.jsp" />
12  </definition>
13
14  <!-- 로그인 페이지 -->
15
16  <definition name="login" template="/WEB-INF/views/Login/project_login.jsp">
17    <put-attribute name="title" value="로그인" />
18    <put-attribute name="body" value="" />
19  </definition>
20
21  <definition name="logout" template="/WEB-INF/views/Login/Logout.jsp">
22    <put-attribute name="title" value="로그아웃" />
23    <put-attribute name="body" value="" />
24  </definition>
25
26  <!-- 메인페이지 -->
27  <definition name="mainpage" extends="baseLayout">
28    <put-attribute name="title" value="메인페이지" />
29    <put-attribute name="content" value="/WEB-INF/views/mainpage/mainPage.jsp" />
30  </definition>
31
```

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"%>
3 <%@ taglib prefix="tiles" uri="http://tiles.apache.org/tags-tiles"%>
4 <!DOCTYPE html>
5 <html>
6 <head>
7   <meta charset="UTF-8">
8   <title><tiles:insertAttribute name="title" /></title>
9 </head>
10
11 <link rel="stylesheet" href="/alphanow/resources/css/tiles/Layout.css">
12
13 <body>
14   <div class="wrap">
15
16     <tiles:insertAttribute name="header" />
17
18     <tiles:insertAttribute name="nav" />
19
20     <div class="Layout_content">
21       <tiles:insertAttribute name="content" />
22     </div>
23
24     <tiles:insertAttribute name="footer" />
25
26   </div>
27 </body>
28 </html>
```

# 재고관리 - 입고관리

 Alphaknow



계시판

공정기준정보

재고관리

생산관리

실적현황

사원관리/등록

입고관리

입고신청목록

전체

결재대기

결재완료

반려

입고완료

입고신청

선택수정

선택삭제

선택반려

결재처리

입고완료

년-월-일

년-월-일

조회

	<input type="checkbox"/>	거래번호	거래처	상태	입고일	총금액	신청일	신청자	수정일	수정자	결재자
1	<input type="checkbox"/>	TC20240429094528	비잉테크	입고완료	2024-04-29 00:46:12.0	48000	2024-04-29 09:45:28.0	허지훈			허지훈
2	<input type="checkbox"/>	TC20240429094516	삼성전자	입고완료	2024-04-29 00:51:48.0	34000	2024-04-29 09:45:16.0	정의건			허지훈
3	<input type="checkbox"/>	TC20240429110935	삼성전자	입고완료	2024-04-29 02:10:14.0	20000	2024-04-29 11:09:35.0	테스트			허지훈


상세


	품목코드	품목명	LOT	입고신청수량	단가	총금액
표시할 내용이 없습니다.						

alphaknow Company MES Web Solution Project

입고관리 페이지입니다.  
입고 신청과 수정, 삭제가 가능하도록 구현하였고 반려 및 결재 후 입고완료할 수 있도록 구현하였습니다.  
입고 신청과 같은 버튼에는 Ajax를 사용하였습니다.

# 재고관리 - 보유재고관리





계시판

공정기준정보

재고관리

생산관리

실적현황

사원관리/등록

재고관리

보유재고현황

년-월-일

년-월-일

검색

	품목코드	품목타입	품목명	현재고(EA)
1	CPU-A01	부품	CPU PCB	90
2	AC-E05	부품	Audio Codec	60


상세


	품목코드	품목명	LOT	현재고(EA)
1	CPU-A01	CPU PCB	PD20240429004526LOT	50
2	CPU-A01	CPU PCB	PD20240429020957LOT	40

alphaknow Company MES Web Solution Project

보유재고관리 페이지입니다.  
입고된 품목의 수량만큼 더해져  
현재고에 반영됩니다.  
품목을 클릭하면 Ajax를 통해 아  
래 lot별로 상세 품목이 정렬되게  
끔 구현하였습니다.

# 생산관리 - 생산계획관리





계시판

공정기준정보

재고관리

생산관리

실적현황

사원관리/등록

생산계획관리

계획물리보기

추가

수정

삭제

<input type="checkbox"/>	생산계획코드	품목코드	재품명	납품처	납품요구수량	생산계획수량	예상보유재고	작수일	완료일	▲
<input type="checkbox"/>	PPC12	214213213	아이템1	휴민테크	1000	2000	1000	2024-02-25	2024-03-21	
<input type="checkbox"/>	PPC26	1234	아이템릴릴	123테크	1010100	213123	13213	2011-01-01	2012-01-01	
<input type="checkbox"/>	PPC45	1234	아이아이	214213	214	214	214	2024-03-07	2024-03-22	
<input type="checkbox"/>	PPC2024041614	11111	123123	123123	123123	123123	123123	2024-04-15	2024-04-17	
<input type="checkbox"/>	PPC2024042508	1111	제품	삼송	300	200	100	2024-04-15	2024-04-24	▼

alphaknow Company MES Web Solution Project

생산계획관리 페이지입니다.  
MES에서의 생산 계획을 수립하  
는 단계를 구성하였습니다.  
추가, 수정 및 삭제가 가능합니다.

# 실적현황

[계시판](#)[공정기준정보](#)[재고관리](#)[생산관리](#)[실적현황](#)[사원관리/등록](#)

## 생산실적현황

2023년 ▾ pcb명 ▾ 계획량 / 실생산량 ▾ 찾기




1월 ▾

1주차 2주차 3주차 4주차 5주차

기준일	자재명	계획량	실생산량	달성률	불량률

생산실적현황 페이지입니다.  
chart.js를 이용하여 그래프를 표  
시하였습니다. 생산실적을 한눈  
에 보기 쉽도록 ui를 구상하였습  
니다.

# 사원관리 / 등록



계시판    공정기준정보    재고관리    생산관리    실적현황    **사원관리/등록**

사원관리

사원등록

	사원번호	사원명	근속현황
1	10000	정의건	재직
2	10005	이민영	신규

사원정보

X

사원번호	10005
사원명	이민영
휴대전화	01078964521
입사일	2024-04-25
아이디	user
비밀번호	1234
부서	생산
직급	인턴
근속현황	신규

수정

삭제

상세정보

상세정보

alphaknow Company MES Web Solution Project

사원관리 / 등록 페이지입니다.  
신규로 입사하거나 퇴사하는 사  
원 또는 정보가 변동되는 사원을  
기록하도록 구현하였습니다.  
상세정보나 수정완료 등은 Ajax  
를 통해 구현하였습니다.

## 주요 사용 기능 - lombok

```
<!-- lombok -->
<dependency>
  <groupId>org.projectlombok</groupId>
  <artifactId>lombok</artifactId>
  <version>1.18.32</version>
  <scope>provided</scope>
</dependency>
```

```
1 package com.spring.alphaknow.dto.rmDTO;
2
3 import java.sql.Date;
4
5 @Getter
6 @Setter
7 @ToString
8 public class ReceivingManagementDTO {
9     int trade_seq;
10    String trade_code;
11    String company_name;
12    String sign_status;
13    Date receiving_date;
14    int product_all_price;
15    Date request_date;
16    String request_person;
17    String modify_request_person;
18    Date modify_request_date;
19    String sign_person;
20    String product_code;
21    String product_name;
22    String LOT;
23    String product_amount;
24    int after_product_amount;
25    int request_amount;
26    int product_price;
27 }
28
```

lombok 을 활용하여 getter setter  
등의 코드를 하드코딩 없이 편하  
게 작업하였습니다.



## 주요 사용 기능 - mybatis

```
<!-- Mybatis -->
<dependency>
  <groupId>commons-beanutils</groupId>
  <artifactId>commons-beanutils</artifactId>
  <version>1.8.0</version>
</dependency>
<dependency>
  <groupId>commons-dbcp</groupId>
  <artifactId>commons-dbcp</artifactId>
  <version>1.2.2</version>
</dependency>
<dependency>
  <groupId>cglib</groupId>
  <artifactId>cglib-nodep</artifactId>
  <version>2.2</version>
</dependency>
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>

  <!-- tag(node) 의 순서에 주의 -->
  <settings>
    <!-- 전달인자로 들어가는 null 이 예러가 나지 않게 함 (전달인자로 null을 줄 수 있음) -->
    <setting name="jdbcTypeForNull" value="NULL" />
  </settings>

  <!-- mapper에서 사용할 타입이 길어서 짧게 별칭 지정 -->
  <typeAliases>
    <typeAlias type="com.spring.alphaknow.dto.ppmDTO.ProductPlanManagementDTO"
      alias="ppmDTO" />

    <typeAlias type="com.spring.alphaknow.dto.employeeDTO.EmployeeDTO"
      alias="employeeDTO" />

    <typeAlias type="com.spring.alphaknow.dto.equipmentDTO.Equipment"
      alias="Equipment" />

    <typeAlias type="com.spring.alphaknow.dto.processcodeDTO.ProcessCodeDTO"
      alias="ProcessCodeDTO" />

    <typeAlias type="com.spring.alphaknow.dto.boardDTO.BoardDTO"
      alias="boardDTO" />
  </typeAliases>
</configuration>
```

mybatis를 이용하여 브라우저와  
데이터베이스를 연동하였습니다.

## 주요 사용 기능 - mybatis

```
1 <?xml version="1.0" encoding="UTF-8" ?>
2 <!DOCTYPE mapper
3     PUBLIC "-//mybatis.org/DTD Mapper 3.0/EN"
4     "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
5 <mapper namespace="com.spring.alphaknow.dao.rmDAO.ReceivingManagementDAO">
6
7     <!-- 첫페이지 -->
8     <select id="receivingManagementSelect" resultType="map">
9         SELECT
10             rm.TRADE_CODE,
11             rc1.COMPANY_NAME,
12             rm.SIGN_STATUS,
13             rm.RECEIVING_DATE,
14             SUM(rm.PRODUCT_ALL_PRICE) PRODUCT_ALL_PRICE,
15             rm.REQUEST_DATE,
16             rm.REQUEST_PERSON,
17             rm.MODIFY_REQUEST_DATE,
18             rm.MODIFY_REQUEST_PERSON,
19             rm.SIGN_PERSON
20         FROM
21             RECEIVING_MANAGEMENT rm
22         JOIN
23             COMPANY_AND_PRODUCT_TEMP capt
24         ON
25             rm.COMPANY_AND_PRODUCT_TEMP_SEQ = capt.COMPANY_AND_PRODUCT_TEMP_SEQ
26         JOIN
27             RECEIVING_COMPANY_LIST rc1
28         ON
29             capt.COMPANY_SEQ = rc1.COMPANY_SEQ
30         JOIN
31             PRODUCT_TEMP nt
```

```
1 package com.spring.alphaknow.dao.rmDAO;
2
3 import java.util.List;
4
5 @Mapper
6 public interface ReceivingManagementDAO {
7     List<ReceivingManagementDTO> receivingManagementSelect();
8     List<ReceivingManagementAjaxDTO> receivingManagementAjaxSelect();
9     List<ReceivingManagementAjax2DTO> receivingManagementAjaxSelect2(String company_seq);
10    void receivingManagementInsert(ReceivingManagementInsertDTO dto);
11    void receivingManagementDelete(ReceivingManagementDTO dto);
12    List<ReceivingManagementAjax3DTO> receivingManagementAjaxSelect3(String trade_code);
13    List<ReceivingManagementAjax4DTO> receivingManagementAjaxSelect4(String trade_code);
14    void receivingManagementUpdate(ReceivingManagementUpdateDTO dto);
15    void receivingSign(ReceivingSignDTO dto);
16    void receivingSign2(ReceivingSignDTO dto);
17    void receivingToInventory(ReceivingManagementAjax3DTO dto);
18    void receivingToInventory2(ReceivingManagementAjax3DTO dto);
19 }
20
```

## Mapper 인터페이스 활용

## 주요 사용 기능 - log4j

```
Console x Progress Problems Search
Tomcat v9.0 Server at localhost [Apache Tomcat/9.0.23@bin\java.exe (2024.5.28. 오전 1:07:04)]

rw.MULTIPLY_REQUEST_DATE,
rw.MODIFY_REQUEST_PERSON,
rw.SIGN_PERSON

8 03:10:24,732 INFO [jdbc.resultsettable(108)] |-----|-----|-----|-----|-----|-----|-----|-----|
8 03:10:24,733 INFO [jdbc.resultsettable(108)] |TRADE_CODE|COMPANY_NAME|SIGN_STATUS|RECEIVING_DATE|PRODUCT_ALL_PRICE|REQUEST_DATE|REQUEST_PERSON|MODIFY_REQ|
8 03:10:24,733 INFO [jdbc.resultsettable(108)] |-----|-----|-----|-----|-----|-----|-----|-----|
8 03:10:24,733 INFO [jdbc.resultsettable(108)] |TC20240429094528|비잉테크|입고완료|2024-04-29 00:46:12.0|48000|2024-04-29 09:45:28.0|허지훈|[null]
8 03:10:24,734 INFO [jdbc.resultsettable(108)] |TC20240429094516|삼성전자|입고완료|2024-04-29 00:51:48.0|34000|2024-04-29 09:45:16.0|정의권|[null]
8 03:10:24,734 INFO [jdbc.resultsettable(108)] |TC20240429110935|삼성전자|입고완료|2024-04-29 02:10:14.0|20000|2024-04-29 11:09:35.0|텍스트|[null]
8 03:10:24,734 INFO [jdbc.resultsettable(108)] |-----|-----|-----|-----|-----|-----|-----|-----|
COMPANY_NAME=비잉테크, REQUEST_PERSON=허지훈, SIGN_PERSON=허지훈, REQUEST_DATE=2024-04-29 09:45:28.0, RECEIVING_DATE=2024-04-29 00:46:12.0, TRADE_CODE=TC20240429094528, SIGN_STATUS=입고완료, P

8 03:10:27,020 INFO [jdbc.sqlonly(74)] SQL : SELECT
CAPT.COMPANY_AND_PRODUCT_TEMP_SEQ,
PT.PRODUCT_CODE,
PT.PRODUCT_NAME,
RCL.COMPANY_SEQ,
RCL.COMPANY_NAME,
PT.PRODUCT_PRICE,
I.PRODUCT_AMOUNT
FROM
PRODUCT_TEMP pt
JOIN
COMPANY_AND_PRODUCT_TEMP capt
ON
PT.PRODUCT_SEQ = CAPT.PRODUCT_SEQ
JOIN
RECEIVING_COMPANY_LIST rcl
ON
CAPT.COMPANY_SEQ = RCL.COMPANY_SEQ
LEFT OUTER JOIN
INVENTORY i
ON
i.PRODUCT_SEQ = pt.PRODUCT_SEQ
8 03:10:27,083 INFO [jdbc.resultsettable(108)] |-----|-----|-----|-----|-----|-----|-----|-----|
8 03:10:27,084 INFO [jdbc.resultsettable(108)] |COMPANY_AND_PRODUCT_TEMP_SEQ|PRODUCT_CODE|PRODUCT_NAME|COMPANY_SEQ|COMPANY_NAME|PRODUCT_PRICE|PRODUCT_AMOUNT|
8 03:10:27,084 INFO [jdbc.resultsettable(108)] |-----|-----|-----|-----|-----|-----|-----|-----|
8 03:10:27,085 INFO [jdbc.resultsettable(108)] |10000|CPU-A01|CPU PCB|10000|삼성전자|500|40|
8 03:10:27,086 INFO [jdbc.resultsettable(108)] |10004|AC-E05|Audio Codec|10003|비잉테크|800|60|
8 03:10:27,086 INFO [jdbc.resultsettable(108)] |10000|CPU-A01|CPU PCB|10000|삼성전자|500|50|
8 03:10:27,086 INFO [jdbc.resultsettable(108)] |10001|PMIC-B02|Sub PMIC|10000|삼성전자|300|[null]|
8 03:10:27,086 INFO [jdbc.resultsettable(108)] |10002|PMIC-B02|Sub PMIC|10000|삼성전자|300|[null]|
```

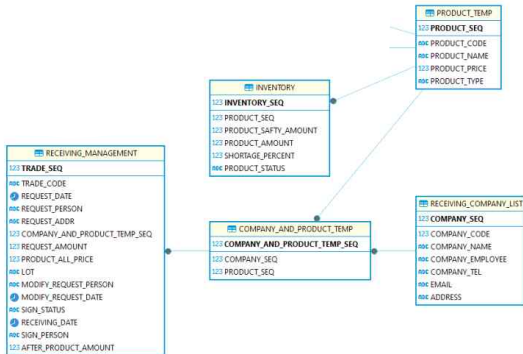
log4j를 통해 실행되는 sql  
문을 확인하며 작업했습니다.

## 주요 사용 기능 - jquery & ajax

```
1 // 클릭한 행의 사원번호 찾기
2*(function () {
3     console.log("test")
4     $(".detailInfo").off("click").on("click", function () {
5         // 사원번호 확보
6         let tr = $(this).parent().parent();
7         // console.log($(this).parent().parent()[0])
8         let empno = tr.find(".selectEmployeeKey").val();
9         // console.log(tr.find(".selectEmployeeKey")[0])
10        // console.log("empno : " + empno);
11        // 사원번호를 들고 아작스로 이동
12
13        // Ajax
14*(function () {
15            // AJAX 요청 실행
16*(function(){
17            url: "/alphaknow/employee/ajax.doSelect?empno=" + empno, // 요청할 URL
18            method: "GET", // HTTP 요청 메서드 (GET, POST 등)
19            dataType: "json", // 응답 데이터 타입 (json, xml, html 등)
20            success: function (data) {
21                // 요청 성공 시 처리할 로직
22                // console.log("data : ", data[0]);
23                // data값 활용
24
25                // 타임스탬프를 Date 객체로 변환
26                let timestamp = data[0].employeeDate;
27                let date = new Date(timestamp);
28
29                // "yyyy-mm-dd" 형식으로 날짜 문자열 포맷팅
30                let formattedDate = formatDate(date);
31
32                // console.log("Formatted Date:", formattedDate);
33
34                // 날짜를 "yyyy-mm-dd" 형식으로 포맷팅하는 함수
35*(function formatDate(date) {
36                let year = date.getFullYear();
37                let month = ('0' + (date.getMonth() + 1)).slice(-2); // 월은 0부터 시작하므로 +1 필요
38                let day = ('0' + date.getDate()).slice(-2);
39
40                return year + '-' + month + '-' + day;
41
```

사원관리 페이지에서  
jquery와 ajax를 활용하여  
데이터값을 불러오는 방식  
도 사용해 보았습니다.

## 주요 사용 기능 - oracleDB



거래처 리스트와 물품 리스트의 키값을 조합하여 거래처가 관리하는 물품 테이블을 만든 후 입고 페이지와 보유재고 페이지에 활용하였습니다.

## 소감

프론트엔드부터 백엔드까지 그리고 프로젝트 설계부터 구현까지 모든 부분을 경험할 수 있는 귀중한 시간이었습니다. 팀과 함께 하나의 목표를 달성하기 위해서 프로젝트를 진행하는것도 매우 재미있었습니다. 기본적인 CRUD를 통한 웹 개발을 경험할 수 있었고 평소 약했던 Ajax나 javascript도 많이 연습할 수 있었습니다. 다만 시간 관리 부분이나 구상 단계에서의 정교함에서 많이 부족함을 느꼈고 차후 프로젝트에서 보완하도록 노력할 것입니다.