

자바 풀스택 & AI를 활용한 개발자양성 취업과정

(APS플랫폼개발) (K-디지털 트레이닝 과정)

Himedia IT인재개발원

25.05.07~25.10.31

# Wooden

- 상품 생산 / 주문 / 재고 / 판매량 / 판매 수익 통합 관리
- 판매량 예측 시스템



고용노동부



KOREATECH  
직업능력심사평가원

자세한 코딩은 깃허브를 확인해주세요.

<https://github.com/ekdpf3636-jpg/wooden>

WOODEN 시연 영상 링크.

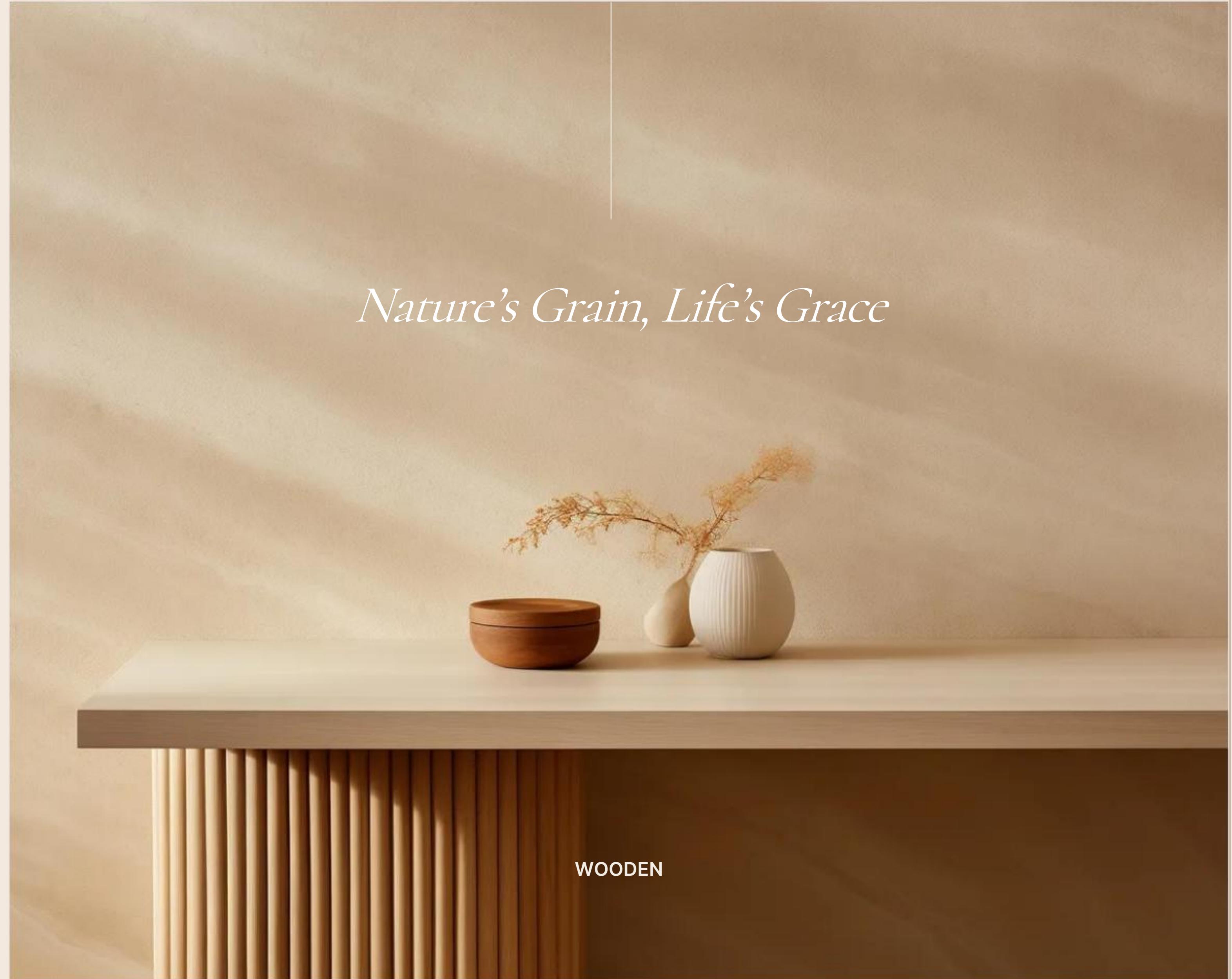
<https://youtu.be/-HoRqU5XT-w>



Premium  
Living / Gifting  
Wood Package

# Contact

1. 개발 목적 및 요구사항
2. 개발환경
3. 역할분담
4. UseCaseDiagram
5. DB구조
6. 프로그램 구조
  - 시스템 기능 흐름
  - 후기





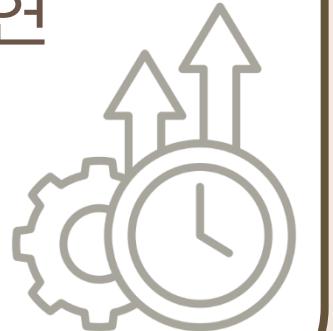
WOODEN

## 관리자의 편의성을 극대화한 시스템

01

실시간 판매량  
통계 / 예측 및  
매출 추이 분석

React-Spring Boot-FastAPI 연동으로  
직관적인 데이터  
시각화 및 실시간  
판매/매출 추이 분석  
기능을 구현



02

생산·재고 등  
데이터 통합 관리

단일 프로세스 내  
생산, 재고, 판매  
통합 관리 및 예측  
기능 구현



03

미래 판매량 예측

미래 판매량을  
예측하여 장기적인  
마케팅/비즈니스  
계획을 세울 수  
있도록 지원



<b>메인 페이지</b>	<p>메인 기본 페이지는 로그인 페이지로 지정 , 비회원은 회원 가입 필요</p> <p>품목별 다음 달 예측치, 당월 판매 수익 및 상품 별 판매 수량 그래프 표시</p> <p>상품과 예측 기간은 드롭 다운으로 선택 가능</p> <p>상단에 각 관리 페이지 (ORDER, BUYER, PLAN, STOCK ) 메뉴 버튼</p> <p>좌측 상단 로고 누르면 메인 페이지로 복귀</p> <p>우측 상단 사용자 이름 표시, 사이트 맵 메뉴 바</p> <p>사용자 이름 표시 밑 로그아웃 버튼</p>	<b>ORDER 공통 요구사항</b>	<p>오더 메뉴 이동 시 판매거래처를 기본 페이지로</p> <p>버튼 : [이전 페이지로 이동, 새로 고침]</p> <p>판매처명 기준 검색</p>
		<b>판매 거래처</b>	<p>판매 거래처 리스트</p> <p>거래처 등록 (판매처명, 담당자, 이메일, 판매처 전화번호, 판매처주소) 입력, 등록완료 후 리스트에 추가</p> <p>판매처명 클릭-&gt;수정, 삭제 폼 열림(새 페이지 이동 X)</p>
		<b>상품 주문서</b>	<p>상품주문서 리스트에 존재 시 해당 판매거래처 삭제 불가</p> <p>주문 완료 처리시 해당 주문서에 연결된 판매거래처 삭제 가능</p>
			<p>상품주문서 리스트</p> <p>상품주문서 등록 : 주문일자&lt;=납품일자 , 판매처 선택(주소 자동입력) , 상품 선택(단가 자동입력), 수량(총 금액 자동계산), 주문/납품 상태는 대기로만 등록 가능</p>
<b>로그인 페이지</b>	<p>비회원은 모든 메뉴 접근 불가능</p> <p>회원가입 후 로그인</p> <p>로그인 후 메인 페이지로 이동</p>	<b>주문완료현황</b>	<p>판매처명 클릭 시 수정, 삭제 폼 열림(새 페이지 이동 X)</p> <p>주문/납품 상태 : [대기] -&gt; [완료] 상태 모두 변경 시 자동으로 주문완료현황 리스트로 이동</p>
<b>회원 가입 페이지</b>	<p>이름 : 한글 30자 내외</p> <p>아이디 : 이메일 형식</p> <p>비밀번호 : 8 자 이상 (더블 체크)</p>		<p>주문이 완료 된 리스트 표시</p> <p>완료된 주문은 수정 삭제 불가능</p>

BUYER 공통 요구사항	바이어 메뉴 이동 시 구매거래처를 기본 페이지로 버튼 : [이전 페이지로 이동, 새로 고침]	PLAN 공통 요구사항	플랜 메뉴 이동 시 상품을 기본 페이지로 버튼 : [이전 페이지로 이동, 새로 고침]
구매 거래처	구매거래처 리스트 거래처 등록 (구매처명, 담당자, 이메일, 구매처 전화번호, 구매처주소) 입력, 등록완료 후 리스트에 추가 구매처명 클릭 시 수정, 삭제 폼 열림(새 페이지 이동 X) 발주 주문서 리스트에 존재 시 해당 구매거래처 삭제 불가 발주 완료 처리시 해당 주문서에 연결된 거래처 삭제 가능 구매처명 기준 검색	상품	상품 리스트 상품 등록 (상품코드, 상품명, 상품규격, 상품단가) 입력, 등록완료 후 리스트에 추가 상품코드 클릭 시 수정, 삭제 폼 열림(새 페이지 이동 X) 상품명 기준 검색
	부품 리스트, 부품명 기준 검색 거래처 등록 (부품명, 구매처명 선택(부품과 연결됨), 부품코드, 부품 규격, 부품 단가) 입력, 등록완료 후 리스트에 추가 부품명 클릭 시 수정, 삭제 폼 열림(새 페이지 이동 X) 부품 발주 주문서 리스트에 존재 시 해당 구매거래처 삭제 불가 발주 완료 처리시 해당 주문서에 연결된 구매거래처 삭제 가능		생산 리스트 생산 등록 (상품명 선택, 생산수량, 생산 상태(생산 중으로만 등록 가능), 생산 시작 <= 생산 종료일) 입력, 등록완료 후 리스트에 추가 상품명 클릭 시 수정, 삭제 폼 열림(새 페이지 이동 X) 생산 상태가 생산완료일 경우 생산 리스트에서 생산완료 재고의 생산완료리스트로 이동
	부품 발주 리스트, 부품명 기준 검색		상품명 기준 검색 BOM이 없는 상품은 생산 불가능
	발주 등록 [구매처 선택(주소 자동입력), 부품명(부품 자동입력) 구매단가(단가 자동입력), 수량(총 금액 자동계산), 입고 상태는 대기로만 등록 가능] 구매처명 클릭 시 수정, 삭제 폼 열림(새 페이지 이동 X) 입고 상태 : [대기] -> [완료] 상태 변경 시 자동으로 STOCK에 입고완료 재고 페이지로 이동		BOM 목록 BOM 등록 [완제품 선택, 부품 선택, 수량(부품 소모량)] 입력, 등록 완료 후 리스트에 추가 완제품명 클릭 시 수정, 삭제 폼 열림(새 페이지 이동 X) 완제품 중복 등록 불가

**STOCK  
공통 요구사항**

스톡 메뉴 이동 시 생산 완료 재고를 기본 페이지로

버튼 : [이전 페이지로 이동, 새로 고침]

상품명 기준 검색

**1. 현재/ 누적 재고 리스트**

상품 페이지에서 추가 된 상품 리스트 추가

**생산 완료 재고****2. 생산완료리스트**

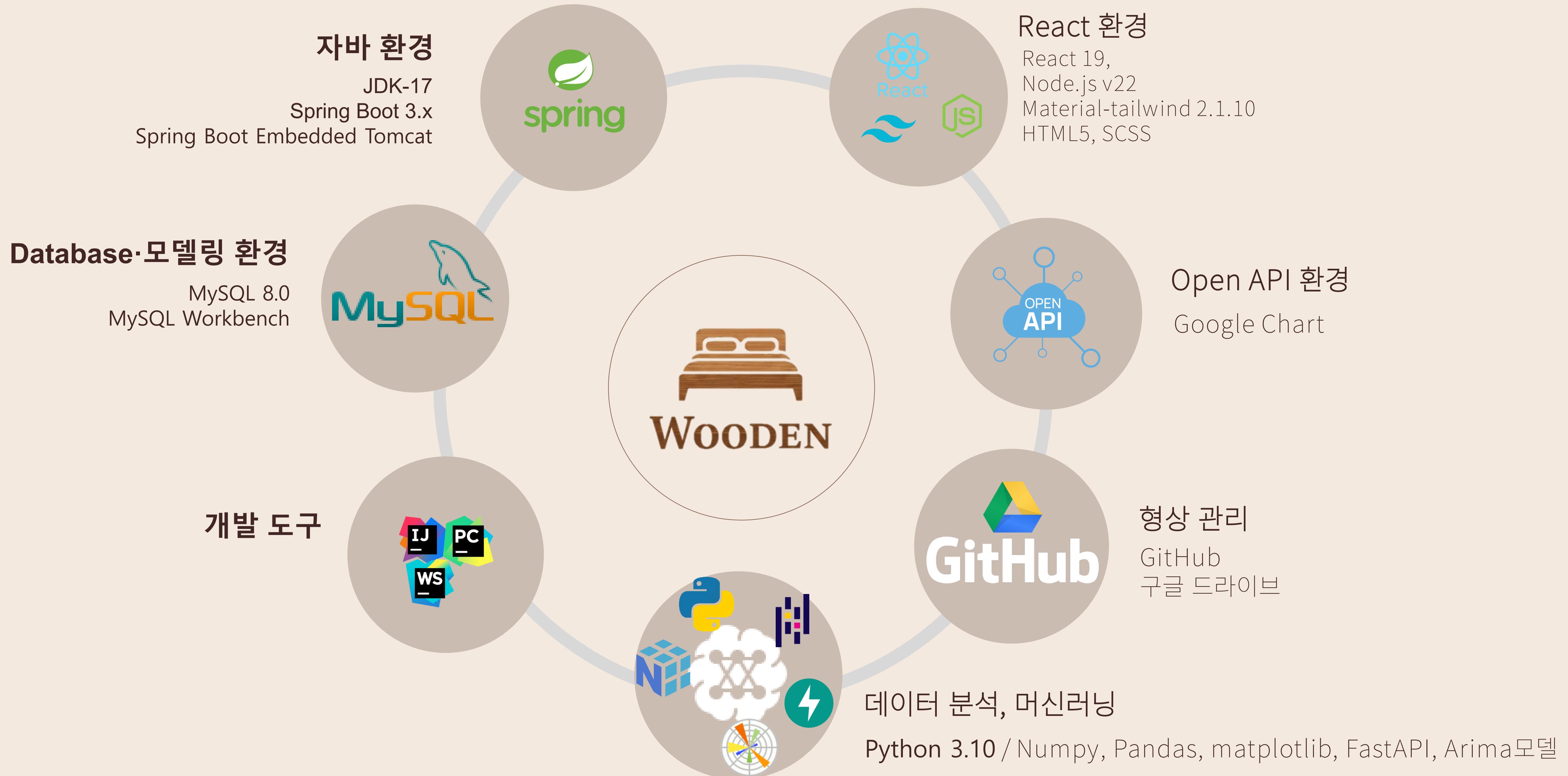
생산 페이지에서 등록된 상품의 상태가 생산완료가 되면 생산 완료 리스트로 이동

**1. 부품 재고 현황**

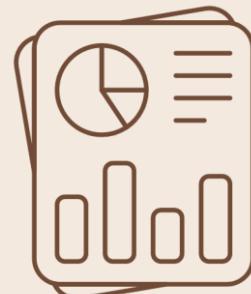
부품 등록 시 부품 재고 현황에 추가

**입고 완료 재고****2. 입고완료 리스트**

부품 발주 등록 후 구매상태가 입고완료가 되면 입고 완료 리스트에 추가



### 3. 역할 분담



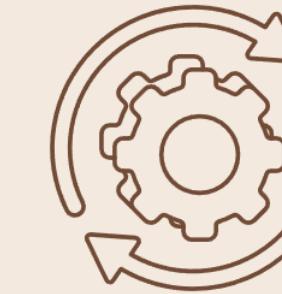
이은지



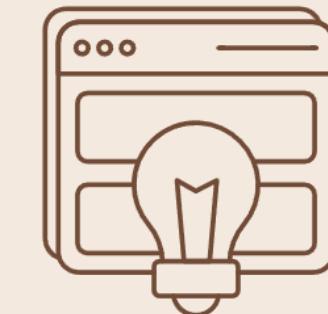
이민호



임대수

팀장  
황다니엘

신동인



김민호



박시온

- 부품  
(Front, Back)
- PPT 작성

- 상품  
(Front, Back)
- MySQL다이어그램
- Use Case
- 구조 명세서
- 사이트맵

- ERP 구조 확립
- 구매 거래처  
(Front, Back)
- Google Chart  
(Front, Back)

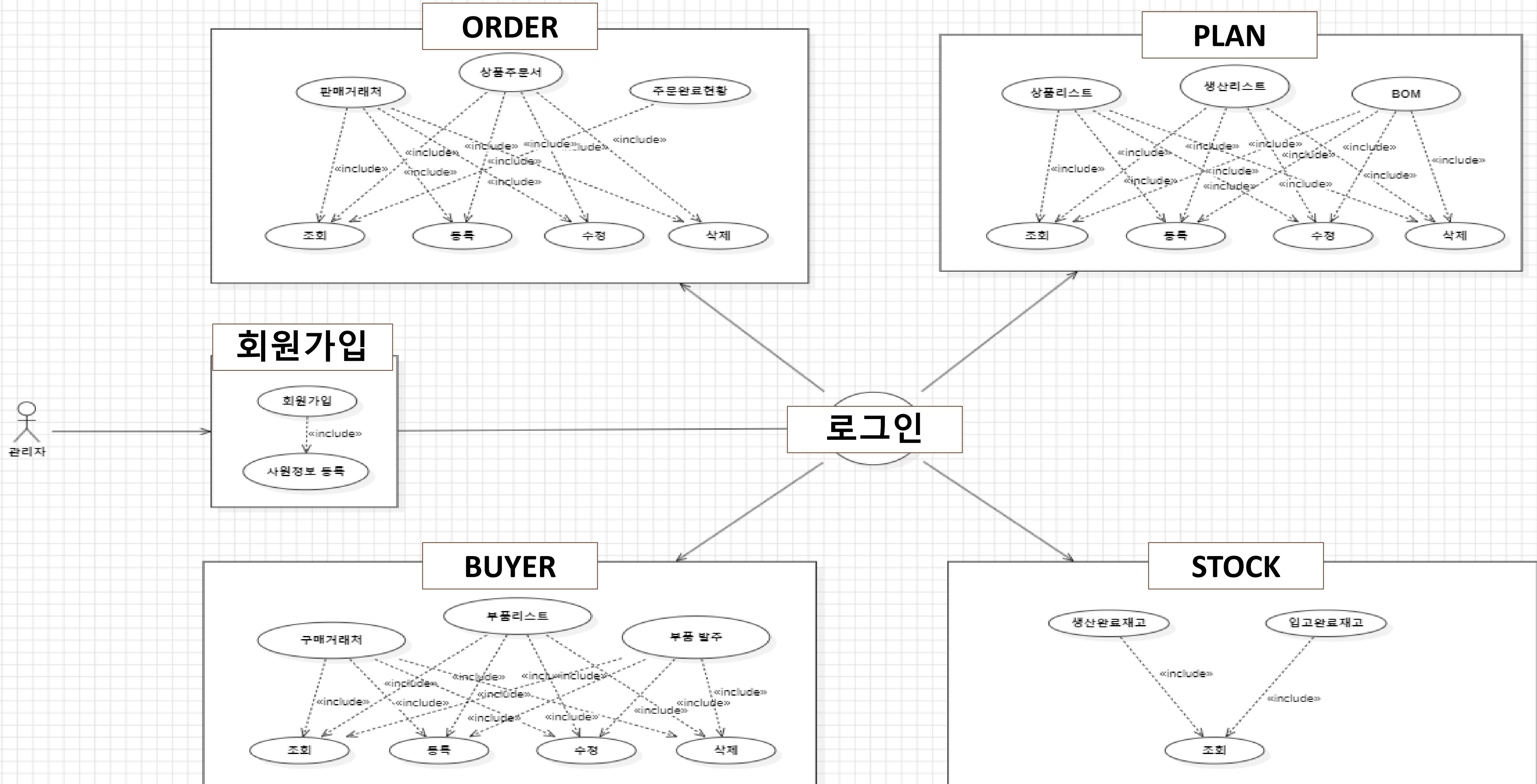
- Wooden 총괄
- 상품 주문서  
(Front, Back)
- 주문 완료 현황  
(Front, Back)
- 생산 완료 재고  
(Front, Back)
- 전체 코딩 리팩토링
- 발표자
- 스토리보드 제작

- Back-end 기초  
빌드
- 생산  
(Front, Back)
- BOM  
(Front, Back)
- 입고 완료 재고  
(Front, Back)
- ARIMA 예측  
(Python)
- 로그인 (Back)

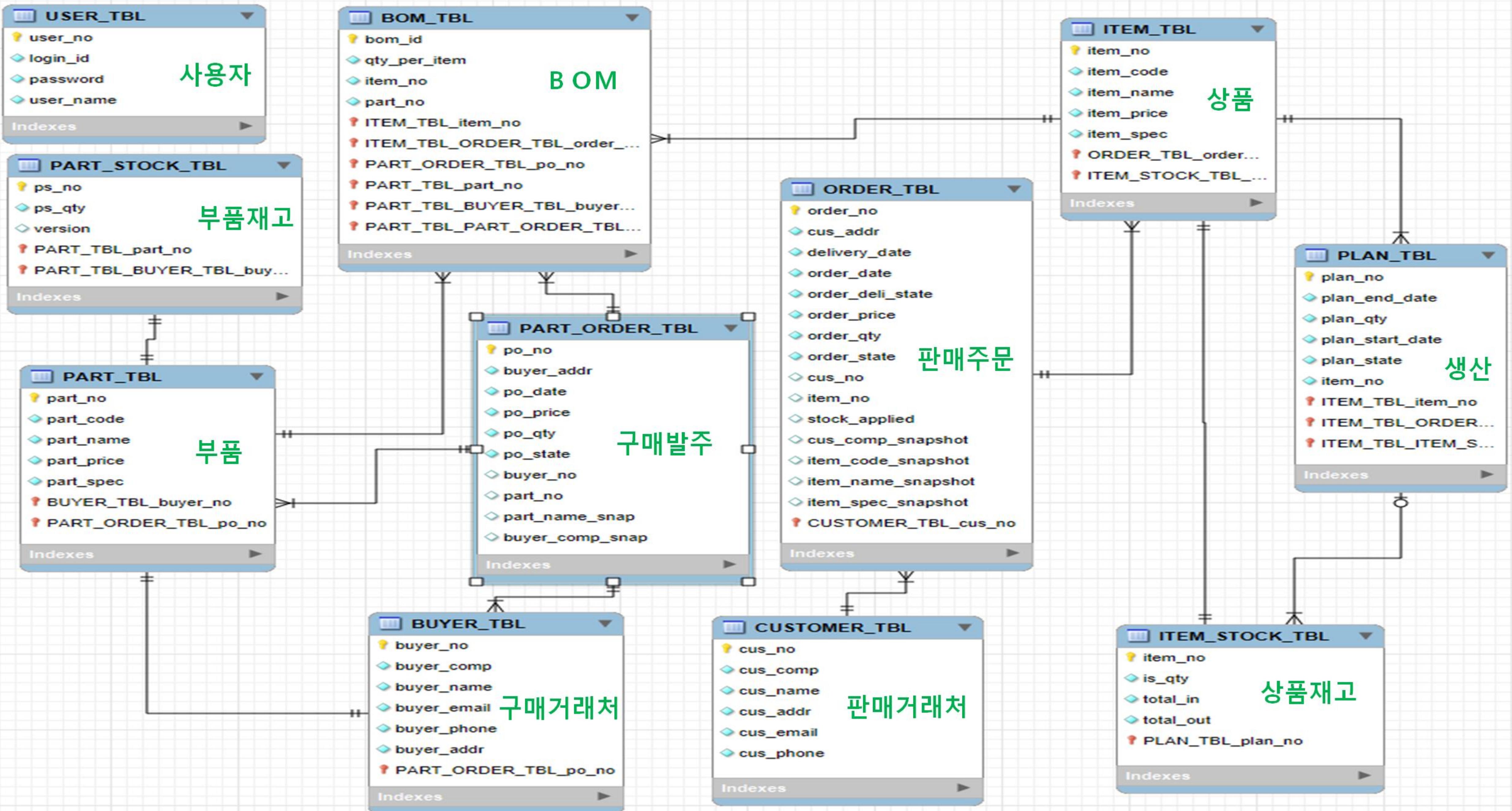
- 부품 발주  
(Front, Back)
- Front-end 전체  
레이아웃 및  
컴포넌트, 툭,  
라우터, 디자인
- 로그인 (Front)
- front 리팩토링

- 판매 거래처  
(Front, Back)
- MySQL다이어그램
- Use Case
- 구조 명세서
- PPT 작성

#### 4 . UseCaseDiagram



# 5 – 1 . DB 모델링



## 5 – 2 . DB테이블 명세서

판매거래처 테이블 CUSTOMER_TBL							
번호	한글명	컬럼명	데이터 타입	크기	Null허용	키	비고
1	판매처 번호	cus_no	INT		N	PK	AUTO_INCREMENT
2	판매처명(FK)	cus_comp	VARCHAR	20	N		
3	판매처 담당자	cus_name	VARCHAR	10	N		
4	판매처 주소	cus_addr	VARCHAR	50	N		
5	판매담당자 이메일	cus_email	VARCHAR	40	N		이메일 @ 형식검사 필요
6	판매처 전화번호	cus_phone	VARCHAR	11	N		정규식 하이픈 포함 X
상품 테이블 ITEM_TBL							
번호	한글명	컬럼명	데이터 타입	크기	Null허용	키	비고
1	일련번호	item_no	INT		N	PK	AUTO_INCREMENT
2	상품코드	item_code	VARCHAR	20	N		
3	상품명	item_name	VARCHAR	30	N		
4	상품단가	item_price	DECIMAL(12,2)		N		모든 가격에 , , 들어감 / 원화 기준 ex) 가로300*세로200*높이150
5	상품규격	item_spec	VARCHAR	40	N		
관리자 테이블 USER_TBL							
번호	한글명	컬럼명	데이터 타입	크기	Null허용	키	비고
1	사원번호	user_no	INT		N	PK	AUTO_INCREMENT
2	로그인용 아이디	login_id	VARCHAR	50	N	UNIQUE	아이디 중복 방지 / 이메일형식 로그인
3	비밀번호	password	VARCHAR	60	N		BCrypt (60자)
4	사원명	user_name	VARCHAR	30	N		
상품 주문 테이블 ORDER_TBL							
번호	한글명	컬럼명	데이터 타입	크기	Null허용	키	비고
1	주문번호	order_no	INT		N	PK	AUTO_INCREMENT
2	판매처 주소	cus_addr	VARCHAR	50	N		
3	배송일자	delivery_date	DATE		N		YY-MM-dd
4	주문일자	order_date	DATE		N	FK	YY-MM-dd
5	납품상태	order_deli_state	VARCHAR	10	N	FK	ex) 고객/주문 흐름 (예: 납품대기/납품완료)
6	판매단가	order_price	DECIMAL(14,2)		N		판매단가*수량 화면에 표시
7	판매수량	order_qty	INT		N		
8	판매상태	order_state	VARCHAR	10	N	FK	ex) 고객/주문 흐름 (예: 판매대기/판매완료)
9	판매처명(FK)	cus_no	INT		FK		셀렉트 컬럼 : cus_comp
10	상품명(FK)	item_no	INT		FK		셀렉트 컬럼 : item_name
11	출고	stock_applied					출고 반영 여부(기준 : 납품완료)
12	판매처명 스냅샷	cus_comp_snapshot	VARCHAR	200			
13	상품코드 스냅샷	item_code_snapshot	VARCHAR	100			
14	상품명 스냅샷	item_name_snapshot	VARCHAR	200			
15	상품규격 스냅샷	item_spec_snapshot	VARCHAR	300			
구매거래처 테이블 BUYER_TBL							
번호	한글명	컬럼명	데이터 타입	크기	Null허용	키	비고
1	구매처 번호	buyer_no	INT		N	PK	AUTO_INCREMENT
2	구매처명	buyer_comp	VARCHAR	20	N		
3	구매처 담당자	buyer_name	VARCHAR	10	N		
4	구매 담당자 이메일	buyer_email	VARCHAR	40	N		이메일 @ 형식검사
5	구매처 전화번호	buyer_phone	VARCHAR	11	N		하이픈 포함 X
6	구매처 주소	buyer_addr	VARCHAR	50	N		

## 5 – 2 . DB테이블 명세서

부품 발주 테이블							PART_ORDER_TBL	
번호	한글명	칼럼명	데이터 타입	크기	Null허용	키	비고	
1	발주번호	po_no	INT		N	PK	AUTO_INCREMENT / part_order = po	
2	구매처 주소	buyer_addr	VARCHAR	100	N			
3	구매입고일자	po_date	DATE		N			
4	구매단가	po_price	DECIMAL(12,2)		N		구매단가*수량 화면에 표시	
5	구매수량	po_qty	INT		N			
6	구매상태	po_state	VARCHAR	20	N		ex) 고객/구매 흐름 (예: 입고대기/입고완료)	
7	구매처명	buyer_no	INT			FK	셀렉트 컬럼 : buyer_comp	
8	부품명(FK)	part_no	INT			FK	part_name	
9	부품명 스냅샷	part_name_snap	VARCHAR	255				
10	구매처명 스냅샷	buyer_comp_snap	VARCHAR	255				
부품 테이블							PART_TBL	
번호	한글명	칼럼명	데이터 타입	크기	Null허용	키	비고	
1	일련번호	part_no	INT		N	PK	AUTO_INCREMENT	
2	부품코드	part_code	VARCHAR	20	N			
3	부품명	part_name	VARCHAR	30	N			
4	부품규격	part_spec	VARCHAR	40	N		ex) 가로300*세로200*높이150	
5	부품단가	part_price	INT		N		모든 가격에 "," 들어감 / 원화 기준	
6	구매처명	buyer_no	INT		N	FK		
생산 테이블							PLAN_TBL	
번호	한글명	칼럼명	데이터 타입	크기	Null허용	키	비고	
1	생산번호	plan_no	INT		N	PK	AUTO_INCREMENT	
2	생산 완료 일자	plan_end_date	DATE		N			
3	생산수량	plan_qty	INT		N			
4	생산 시작 일자	plan_start_date	DATE		N			
5	생산상태	plan_state	VARCHAR	255	N		ex) 흐름 (예: 생산중/생산완료)	
6	상품명(FK)	item_no	INT		N	FK	셀렉트 컬럼 : item_name	
상품 재고 테이블							ITEM_STOCK_TBL	
번호	한글명	칼럼명	데이터 타입	크기	Null허용	키	비고	item_stock = is
1	일련번호	item_no	INT		N	공유 PK	AUTO_INCREMENT	
2	수량	is_qty	INT		N			
3	누적입고	total_in	INT		N			
4	누적출고	total_out	INT		N			
부품재고 테이블							PART_STOCK_TBL	
번호	한글명	칼럼명	데이터 타입	크기	Null허용	키	비고	part_stock = ps
1	부품 재고 일련번호	ps_no	INT		N	PK	AUTO_INCREMENT	
2	수량	ps_qty	INT		N			
3	낙관적 락 버전컬럼	version	INT					
BOM 테이블							BOM_TBL	
번호	한글명	칼럼명	데이터 타입	크기	Null허용	키	비고	
1	BOM ID	bom_id	INT		N	PK	AUTO_INCREMENT	
2	수량	qty_per_item	INT		N		완제품 1개당 필요한 해당 부품 수량	
3	완제품	item_no	INT		N	FK		
4	일련번호	part_no	INT		N	FK		

## 5 – 3 . WOODEN DB 핵심 요약

1. 중심축 : Item	- *Item(상품)*이 모든 흐름의 출발점. ItemStock과 1:1(공유 PK)로 묶어서 완제품 재고를 직결 관리. - is_qty 현재고, total_in 누적입고(=생산완료), total_out 누적출고(=납품완료)
2. 생산 라인	- Plan(생산계획) N:1 → Item - 계획이 완료되면 서비스 레벨에서 ItemStock.produce(qty) 실행해 재고/누적입고 반영. BOM(Item↔Part N:M) - qty_per_item로 완제품당 부품 소요량 고정. 생산 시 부품 차감 로직의 근거.
3. 판매/주문 라인	- Customer 1:N → Order, Order N:1 → Item - 상태 두 축: order_state(승인) + order_deli_state(납품) 둘 다 완료일 때만 ItemStock.sell(qty) 실행해 출고/누적출고 반영. - 스냅샷 필드(거래처명, 품목명/코드/스펙) FK가 바뀌거나 지워져도 과거 문서/리포트가 깨지지 않게 방어.
4. 구매거래처/부품 라인	- Buyer 1:1 → Part(선택적 매피) PartStock 1:1(공유 PK) → Part - 낙관적 락(version)으로 동시 변경 안전장치. - PartOrder N:1 → Buyer, N:1 → Part po_state가 "입고완료"되면 PartStock.changeQty(+qty)로 증가. - 생산 시 BOM 기준으로 PartStock.changeQty(-소요) 차감.
5. 상태 변경 = 재고 반영 트리거	- 생산: Plan 생산완료 → ItemStock.produce - 판매: Order 승인완료 + 납품완료 → ItemStock.sell - 부품 발주: PartOrder 입고완료 → PartStock.changeQty(+) - 생산 소요: BOM 기준 부품 차감 → PartStock.changeQty(-)
6. WOODEN DB설계 의도	- 공유 PK 1:1(Item↔ItemStock, Part↔PartStock) 단순 조인, 강한 무결성, 조회/집계, 비용 예측 가능. - 스냅샷 : 과거 화면/리포트의 불변성 보장. - 명시적 상태 플래그 “언제 재고를 건드릴지”를 코드로 정의해서 데이터 일관성 유지. - BOM 유니크(item_no, part_no) 소요 중복 등록 방지, 계산 신뢰성 확보.
7. 리포트/예측	- 판매 수익 Top10 : Order에서 “승인완료+납품완료”만 결산. - 재고 현황 : ItemStock의 현재/누적 바로 집계. - 예측: 주간 히스토리(빈 주=0) → 파이썬 엔진 → 실적(검정)·예측(파랑) 연속 시각화.

# 6 – 1 . Front-end 구조 명세서

	패키지/폴더	파일명	요청방식	매핑명(URI/Route)	비고
Main 페이지	src/pages	Wooden MainPage.jsx	/		메인 페이지 ( Python ARIMA 예측치 시작화 목적 )
	src/api	forecastAPI.jsx			Python ARIMA
ORDER	src/api	CustomerAPI.jsx	POST/PUT/DELETE	/order/sellercustomer	Axios/ 조회, 등록, 수정, 삭제 기능
	src/form/order	SellCustomerForm.jsx	-		판매 거래처 등록, 수정, 삭제 품
	src/pages/order	SellerCustomerListPage.jsx	GET	/order/sellercustomer	ORDER > 판매거래처 목록 화면
주문	src/api	OrderListAPI.jsx		/order	Axios/ 조회, 등록, 수정, 삭제 기능
	src/form/order	OrderListForm.jsx	POST/PUT/DELETE	-	상품주문서 등록, 수정, 삭제 품
	src/pages/order	OrderListPage.jsx	GET	/order/orderlist	ORDER > 상품주문서 목록 화면
	src/pages/order	OrderApprovePage.jsx	GET	/order/orderreceive	ORDER > 주문 완료 현황 리스트
BUYER	src/api	BuyerListAPI.jsx		/buyer/buyercustomer	Axios/ 조회, 등록, 수정, 삭제 기능
	src/form/buyer	BuyerCustomerForm.jsx	POST/PUT/DELETE	-	구매 거래처 등록, 수정, 삭제 품
	src/pages/buyer	BuyerListPage.jsx	GET	/buyer/buyercustomer	BUYER > 구매거래처 목록 화면
구매	src/api	PartListAPI.jsx		/buyer/partlist	Axios/ 조회, 등록, 수정, 삭제 기능
	src/form/buyer	PartListForm.jsx	POST/PUT/DELETE	-	부품 리스트 등록, 수정, 삭제 품
	src/pages/buyer	PartListPage.jsx	GET	/buyer/partlist	BUYER > 부품리스트 목록 화면
	src/api	PartOrderAPI.jsx		/buyer/partorder	Axios/ 조회, 등록, 수정, 삭제 기능
	src/form/buyer	PartOrderForm.jsx	POST/PUT/DELETE	-	부품 발주 등록, 수정, 삭제 품
	src/pages/buyer	PartOrderListPage.jsx	GET	/buyer/partorder	BUYER > 부품 발주 목록 화면
arrays	패키지/폴더	파일명	요청방식	매핑명(URI/Route)	비고
추가기능	src/arrays	MainArrays.jsx			Nav,Aside에 들어가는 기본적인 배열 정리
	src/arrays	TableArryas.jsx			테이블 FormData 초기화용 배열 정리
	src/arrays	SellerCustomerArrays.jsx			판매거래처 배열 정리
	src/arrays	OrderListArrays.jsx			상품주문서 배열 정리
	src/arrays	BuyerCustomerArrays.jsx			구매거래처 배열 정리
	src/arrays	PartListArrays.jsx			부품리스트 배열 정리
	src/arrays	PartOrderListArrays.jsx			부품 발주 배열 정리
	src/arrays	ItemListArrays.jsx			상품리스트 배열 정리
	src/arrays	PlanListArrays.jsx			생산계획 배열 정리
	src/arrays	BOMListArrays.jsx			BOM 배열 정리
	src/arrays	ItemStockArrays.jsx			상품재고 배열 정리
	src/arrays	PartStockArrays.jsx			부품재고 배열 정리
layouts	페이지/폴더	파일명	요청방식	매핑명(URI/Route)	비고
레이아웃	src/layouts	BasicLayout.jsx			기본 헤더,좌측리스트,우측하단 메인바디부분 렌더링하는 레이아웃
	src/layouts/aside	AsideLayout.jsx			useEvent에서 AsideComponent로 props전달할 때의 aside의 기본 틀
	src/layouts/header	GuideMenuLayout.jsx			사용자 편의 목적 전체 메뉴 레이아웃
	src/layouts/header	HeaderLayout.jsx			페이지 헤더 부분 레이아웃
	src/layouts/header	UserInfoCard.jsx	GET		유저 이름/로그아웃 버튼 레이아웃 관련 담당
components	페이지/폴더	파일명	요청방식	매핑명(URI/Route)	비고
컴포넌트	src/components	AsideComponent.jsx			페이지 좌측 리스트를 URL 경로 따라 렌더링하는 목적인 단일 컴포넌트
	src/components	BackButtonComponent.jsx			ListPage들마다 공통적으로 사용하는 뒤로가기 버튼 컴포넌트
	src/components	ButtonComponent.jsx			ListPage,모달컴포넌트 안에서 공통적으로 사용하는 버튼 컴포넌트
	src/components	CloseBtnComponent.jsx			모달컴포넌트,전체메뉴에서 공통적으로 사용하는 단기버튼 컴포넌트
	src/components	InlineSelectCell.jsx			상품주문서,부품발주 ListPage,Form에서 재사용하는 드롭다운 컴포넌트
	src/components	ModalComponent.jsx			각각의 ListPage들마다 공통적으로 사용하는 모달 컴포넌트
공통 api	페이지/폴더	파일명	요청방식	매핑명(URI/Route)	비고
API	src/api	axios.js			페이지들마다 공통으로 사용하는 AXIOS라이브러리
	src/api	config.js			BackEnd와 연결을 목적으로 하는 공통적인 주소 설정

A	B	C	D	E	F	G	H
PLAN	패키지/폴더	파일명	요청방식	매핑명(URI/Route)	비고		
생산	src/api	ItemListAPI.jsx		/plan/itemlist	Axios/ 조회, 등록, 수정, 삭제 기능		
	src/form/plan	ItemListForm.jsx	POST/PUT/DELETE	-	상품 리스트 등록, 수정, 삭제 품		
	src/pages/plan	ItemListPage.jsx	GET	/plan/itemlist	PLAN > 상품리스트 목록 화면		
STOCK	src/api	PlanListAPI.jsx	GET/POST/PUT/DELETE/PATCH	/plan	Axios/ 조회, 등록, 수정, 삭제 기능		
재고	src/form/plan	PlanListForm.jsx	-	-	생산 리스트 등록, 수정, 삭제 품		
	src/pages/plan	PlanListPage.jsx	GET	/plan/planlist	PLAN > 생산리스트목록 화면		
USER	src/api	BomAPI.jsx		/plan/bomlist	Axios/ 조회, 등록, 수정, 삭제 기능		
	src/form/plan	BomForm.jsx	POST/PUT/DELETE	-	BOM 등록, 수정, 삭제 품		
	src/pages/plan	BomListPage.jsx	GET	/plan/bomlist	PLAN > BOM 목록 화면		
사용자	src/api/user	userAPI.jsx		-	Axios/ 로그인 상태 확인		
	src/pages/user	UserLogin.jsx	POST	/login	사용자 로그인 화면을 띄움		
	src/components/user	LoginComponent.jsx		-	로그인 화면의 기능을 담당하는 컴포넌트		
	src/router/protected	ProtectedRouter.jsx		-	redux를 사용해 loginSlice의 isLoggedIn의 상태가 true면 /으로 이동, false면 /Login으로 이동시키는 역할		
	src/slice	loginSlice.jsx		-	Redux를 통해 로그인 상태 전역 관리, 로그인 성공 시 쿠키를 저장하고, 로그아웃 시 쿠키를 삭제 해 인증 상태를 유지/해제 함.		
	src/util	cookieUtil.jsx		-	React-Cookie로 setCookie/getCookie/removeCookie를 선언, loginSlice에서 쿠키를 조회저장/삭제할 수 있는 기능 담당.		
	src	store.jsx		-	loginSlice를 reducer로 지정함		
	src/api/user	signupAPI.jsx		-	Axios/		
	src/pages/user	UserJoin.jsx	GET/POST	/join	사용자 회원가입 페이지		
hook	페이지/폴더	파일명	요청방식	매핑명(URI/Route)	비고		
훅	src/hooks	useCRUD.jsx			CRUD 관련 상수,함수 모음 / ListPage에서 구조분해할당 목적		
	src/hooks	useEvent.jsx			페이지 이벤트 관련 상수,함수 모음		
SCSS	페이지/폴더	파일명	요청방식	매핑명(URI/Route)	비고		
스타일	src/components	Modal.scss			모달 페이지의 스타일을 담당할 SCSS		
	src/pages/user	UserLogin.scss			로그인 페이지의 스타일을 담당할 SCSS		
	src	App.scss			페이지의 전체적인 기본 스타일링을 설정할 목적인 SCSS		
	src	Responsive1500.scss			노트북화면에 맞춘 기본적인 반응형 디자인 SCSS		
router	페이지/폴더	파일명	요청방식	매핑명(URI/Route)	비고		
라우터	src/router	BuyerRouter.jsx			Nav의 BUYER를 클릭하면 이동할 수 있는 URL을 모아둔 라우터		
	src/router	OrderRouter.jsx			Nav의 ORDER를 클릭하면 이동할 수 있는 URL을 모아둔 라우터		
	src/router	PlanRouter.jsx			Nav의 PLAN을 클릭하면 이동할 수 있는 URL을 모아둔 라우터		
	src/router	StockRouter.jsx			Nav의 STOCK을 클릭하면 이동할 수 있는 URL을 모아둔 라우터		
	src/router	root.jsx			RouterProvider를 사용해 각각의 페이지들마다 URL을 설정하는 기본적인 Router		

## 6 – 2 . Back-end 구조 명세서

구조	구분	패키지 명	파일 이름	요청 방식	매핑 명	비고			
config(설정)	common(공통)	com.wooden.config	CorsConfig.java			CORS 정책 허용 설정	partstock(부품재고)	com.wooden.controller	PartStockController.java
			PasswordConfig.java			비밀번호 암호화(BCrypt) 설정			
			ForecastEngineProps.java			예측 엔진 호출에 쓰는 설정 값"을 외부화해서 보관			
			RestClientConfig.java			HTTP 호출을 담당하는 클라이언트 빙을 만드는 곳			
Controller DTO <-> Service	customer	com.wooden.controller	CustomerController.java	GET	/api/order/sellercustomer	판매거래처 목록 조회	Controller DTO <-> Service	com.wooden.controller	BomController.java
				POST	/api/order/sellercustomer	판매거래처 등록			
				PUT	/api/order/sellercustomer/{cusNo}	판매거래처 수정			
				DELETE	/api/order/sellercustomer/{cusNo}	판매거래처 삭제			
	buyer	com.wooden.controller	BuyerController.java	GET	/api/buyer/buyercustomer	구매거래처 목록 조회			
				POST	/api/buyer/buyercustomer	구매거래처 추가			
				PUT	/api/buyer/buyercustomer/{buyerNo}	구매거래처 수정			
				DELETE	/api/buyer/buyercustomer/{buyerNo}	구매거래처 삭제			
	item	com.wooden.controller	ItemController.java	GET	/api/plan/itemlist	품목 목록 조회			
				GET	/api/plan/itemlist/main	전체 아이템 목록 (드롭다운용)			
				POST	/api/plan/itemlist	품목 등록			
				PUT	/api/plan/itemlist/{itemNo}	품목 수정			
				DELETE	/api/plan/itemlist/{itemNo}	품목 삭제			
controller DTO <-> Service	part	com.wooden.controller	PartController.java	GET	/api/buyer/partlist	부품 목록 조회			
				POST	/api/buyer/partlist	부품 등록			
				PUT	/api/buyer/partlist/{partNo}	부품 수정			
				DELETE	/api/buyer/partlist/{partNo}	부품 삭제			
	order	com.wooden.controller	OrderController.java	GET	/api/order	전체 주문 조회			
				POST	/api/order	주문 등록			
				PUT	/api/order/{orderNo}	주문 수정			
				PATCH	/api/order/{id}/status	주문 상태 변경			
				GET	/api/order/completed	주문 상태 완료 목록			
				DELETE	/api/order/{orderNo}	주문 삭제			
	part_order	com.wooden.controller	PartOrderController.java	GET	/api/buyer/partorder	미완료 목록			
				GET	/api/buyer/partorder/completed	완료 목록			
				POST	/api/buyer/partorder	발주 등록			
				PUT	/api/buyer/partorder/{poNo}	발주 수정			
				DELETE	/api/buyer/partorder/{poNo}	발주 삭제			
Service	plan	com.wooden.controller	PlanController.java	GET	/api/plan/planlist	생산 목록 조회	customer(판매거래처)	com.wooden.service	CustomerService.java
				GET	/api/plan/planlist/completed	완료된 생산계획 조회			
				POST	/api/plan/planlist	생산 등록			
				PUT	/api/plan/planlist/{planNo}	생산 수정			
				PATCH	/api/plan/planlist/{planNo}/status	생산 상태 전환			
				DELETE	/api/plan/planlist/{planNo}	생산 삭제			
	itemstock(상품재고)	com.wooden.controller	ItemStockController.java	GET	/api/stock/item	생산완료리스트+상품재고표시	buyer(구매거래처)	com.wooden.service	BuyerService.java
				GET	/api/stock/item/{itemNo}	생산 완료 반영 (현재 재고/ 누적 입고 증가)			
				POST	/api/stock/item/adjust	주문 완료 반영 (현재 재고 감소/ 누적 출고 증가)			

## 6 – 2 . Back-end 구조 명세서

구조	구분	패키지 명	파일 이름	요청 방식	매핑 명	비고	구조	구분	패키지 명	파일 이름	요청 방식	매핑 명	비고
service	item(상품)	com.wooden.service	ItemService.java			상품 서비스 인터페이스	dto	customer(판매거래처)	com.wooden.dto	CustomerRequestDto.java			요청 DTO
			ItemServiceImpl.java			상품 서비스 구현체 (상품 등록, 조회, 수정, 삭제)				CustomerResponseDto.java			응답 DTO
	part(부품)	com.wooden.service	PartService.java			부품 서비스 인터페이스		buyer(구매거래처)	com.wooden.dto	BuyerRequestDto.java			구매처 요청 DTO
			PartServiceImpl.java			부품 서비스 구현체(부품 등록, 조회, 수정, 삭제)				BuyerResponseDto.java			구매처 응답 DTO
	order(상품 주문)	com.wooden.service	OrderService.java			주문 서비스 인터페이스		item(상품)	com.wooden.dto	ItemRequestDto.java			품목 요청 DTO
			OrderServiceImpl.java			주문 서비스 구현체 (주문 등록, 상태값변경)				ItemResponseDto.java			품목 응답 DTO
	partorder(부품 발주)	com.wooden.service	PartOrderService.java			부품 발주 서비스 인터페이스		part(부품)	com.wooden.dto	PartRequestDto.java			부품 요청 DTO
			PartServiceImpl.java			부품 발주 서비스 구현체 (발주 생성, 입고 처리 등)				PartResponseDto.java			부품 응답 DTO
	plan(생산)	com.wooden.service	PlanService.java			생산원료재고 서비스 인터페이스		order(상품 주문)	com.wooden.dto	OrderRequestDto.java			주문 요청 DTO
			PlanServiceImpl.java			생산원료재고 서비스 구현체 (생산 등록)				OrderResponseDto.java			주문 응답 DTO
	itemstock(상품재고)	com.wooden.service	ItemStockService.java			상품 재고 서비스 인터페이스		order(상품 주문)	com.wooden.dto	OrderStatusUpdateDto.java			주문 상태 부분 업데이트 DTO
			ItemStockServiceImpl.java			상품 재고 서비스 구현체 (조회)				PartOrderRequestDto.java			주문 발주 요청 DTO
	partstock(부품재고)	com.wooden.service	PartStockService.java			부품 재고 서비스 인터페이스		partorder(부품 발주)	com.wooden.dto	PartOrderResponseDto.java			주문 발주 응답 DTO
			PartStockServiceImpl.java			부품 재고 서비스 구현체 (조회)				PlanRequestDto.java			생산 요청 DTO
	bom	com.wooden.service	BomService.java			BOM 서비스 인터페이스		itemStock(상품재고)	com.wooden.dto	PlanResponseDto.java			생산 응답 DTO
			BomServiceImpl.java			BOM 서비스 구현체 (등록, 조회, 수정, 삭제)				ItemStockRequestDto.java			재고 증감 요청 DTO
MainChart	com.wooden.service		ForecastEngineClient.java			예측 엔진 클라이언트(POST 호출)		itemStock(상품재고)	com.wooden.dto	ItemStockResponseDto.java			재고 증감 응답 DTO
			ForecastService.java			예측 서비스 인터페이스				PartStockRequestDto.java			부품 재고 증감 요청 DTO
			ForecastServiceImpl.java			예측 서비스 구현체(히스토리→외부예측→시계열 DTO)		PartStock(부품재고)	com.wooden.dto	PartStockResponseDto.java			부품 재고 증감 응답 DTO
			WeeklyHistoryService.java			주간 히스토리 조회 인터페이스				BomRequestDto.java			BOM 요청 DTO
			WeeklyHistoryServiceImpl.java			주간 히스토리 구현체		bom	com.wooden.dto	BomResponseDto.java			BOM 응답 DTO
			SalesService.java			납품원료 기준 매출·수량 집계 서비스				GlobalExceptionHandler.java			전역 예외 처리
	User	com.wooden.service	LoginService.java			인증 서비스 인터페이스				GlobalExceptionHandler.java			
			LoginServiceImpl.java			인증 서비스 구현체				GlobalExceptionHandler.java			

## 6 – 2 . Back-end 구조 명세서

구조	구분	패키지 명	파일 이름	요청 방식	매핑 명	비고
dto	MainChart	com.wooden.dto	ForecastSeriesDto.java			주간 시계열 응답 DTO
			WeeklyHistoryDto.java			주간 실적 시계열 DTO
User	User	com.wooden.dto	LoginRequestDto.java			로그인 요청 DTO
			LoginResponseDTO.java			로그인 응답 DTO
			SignupRequestDto.java			회원가입 요청 DTO
			UserResponseDto.java			회원정보 응답 DTO
repository	customer	com.wooden.repository	CustomerRepository.java	Long		고객 리포지토리
	buyer	com.wooden.repository	BuyerRepository.java	Long		구매처 리포지토리
	item	com.wooden.repository	ItemRepository.java	Long		품목 리포지토리
	part	com.wooden.repository	PartRepository.java	Long		부품 리포지토리
	order	com.wooden.repository	OrderRepository.java	Long		주문 리포지토리
	part_order	com.wooden.repository	PartOrderRepository.java	Long		부품 발주 리포지토리
	plan	com.wooden.repository	PlanRepository.java	Long		생산 리포지토리
	itemstock	com.wooden.repository	ItemStockRepository.java	Long		완제품 재고 리포지토리
	partstock	com.wooden.repository	PartStockRepository.java	Long		부품 재고 리포지토리
	bom	com.wooden.repository	BomRepository.java	Long		BOM 리포지토리
mainchart	MainChart	com.wooden.repository	UserRepository.java	Long		회원 리포지토리
			SalesRepository.java	String		메인차트 집계 JDBC 리포지토리

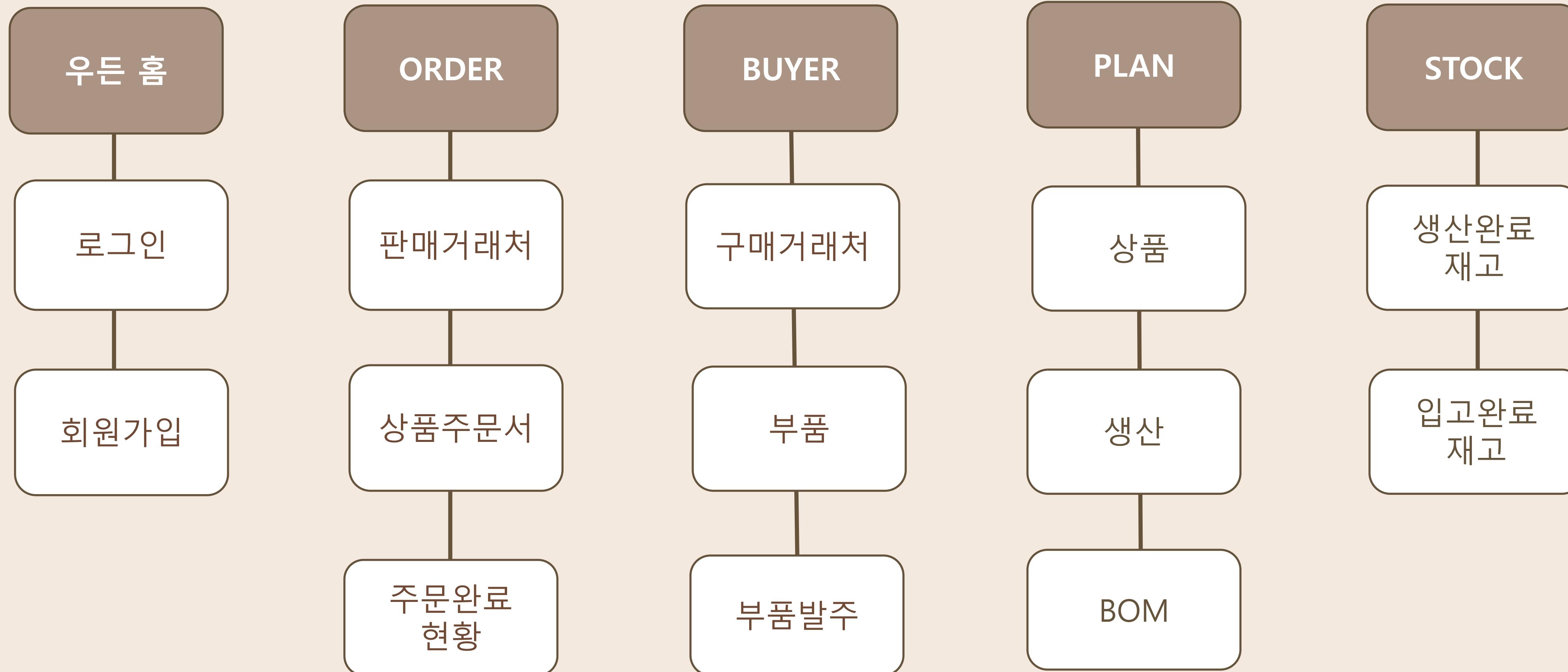
구조	구분	패키지 명	파일 이름	요청 방식	매핑 명	비고
entity	Customer(판매거래처)	com.wooden.domain	Customer			CUSTOMER_TBL
	Buyer(구매거래처)	com.wooden.domain	Buyer			BUYER_TBL
	Item(상품)	com.wooden.domain	Item			ITEM_TBL
	Part(부품)	com.wooden.domain	Part			PART_TBL
	BOM	com.wooden.domain	BOM			BOM_TBL (bom_id)
	Order(상품주문)	com.wooden.domain	Order			ORDER_TBL (order_no)
	PartOrder(부품발주)	com.wooden.domain	PartOrder			PART_ORDER_TBL (po_no)
	Plan(생산완료재고)	com.wooden.domain	Plan			PLAN_TBL (plan_no)
	ItemStock(상품재고)	com.wooden.domain	ItemStock			ITEM_STOCK_TBL (is_no)
User	PartStock(부품재고)	com.wooden.domain	PartStock			PART_STOCK_TBL (ps_no)
	User	com.wooden.domain	User			USER_TBL (user_no)

# - 시스템 기능 흐름

- 사이트 맵
- Front-end 구조
- Back-end 구조
- 공통흐름
- JOIN
- LOGIN
- MAIN
- ORDER
- BUYER
- PLAN
- STOCK

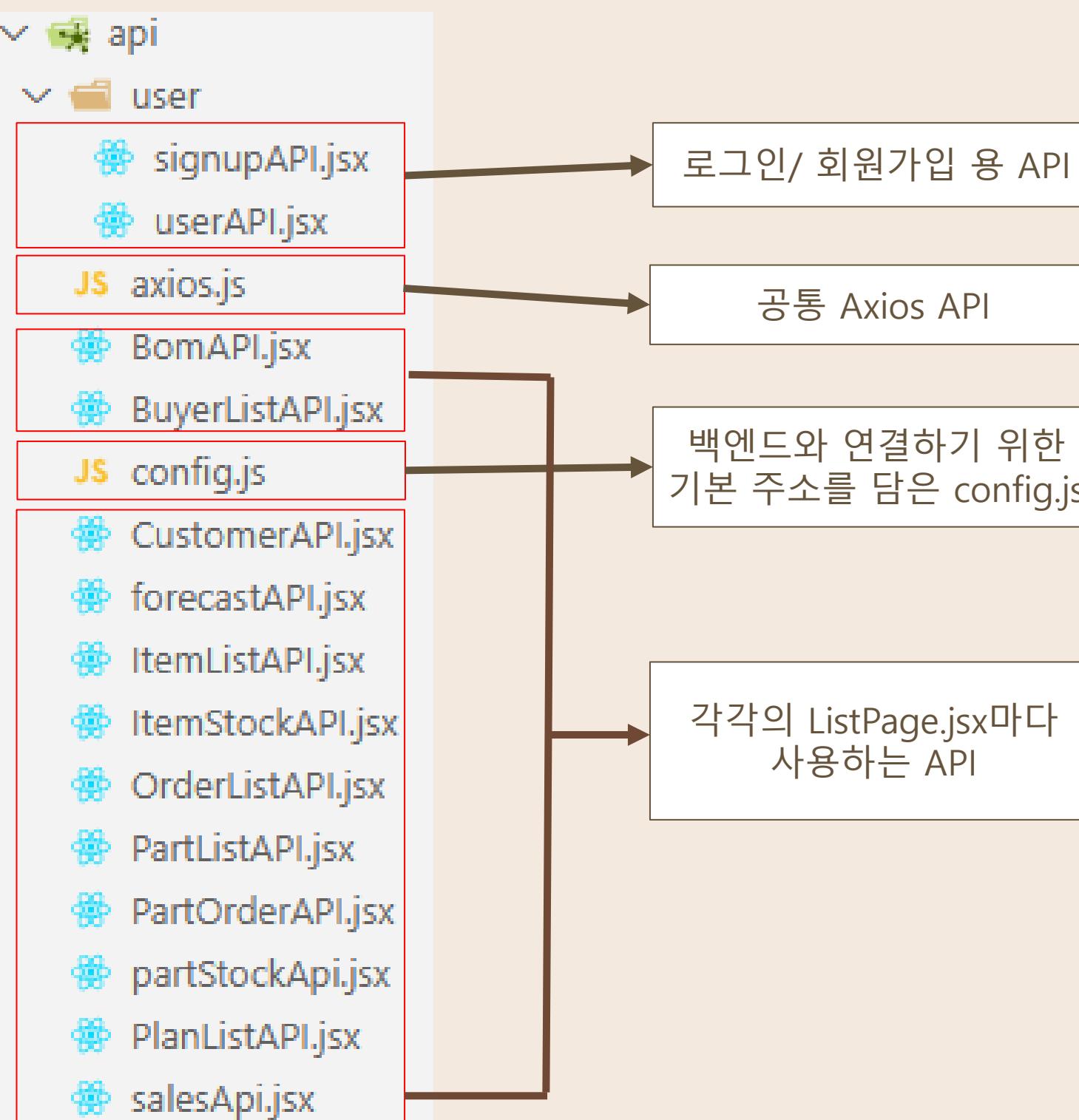


# 사이트 맵



# Front-end 구조

페이지들마다  
조회/ 등록/ 수정/ 삭제 기능을  
사용할 수 있게 하는 API



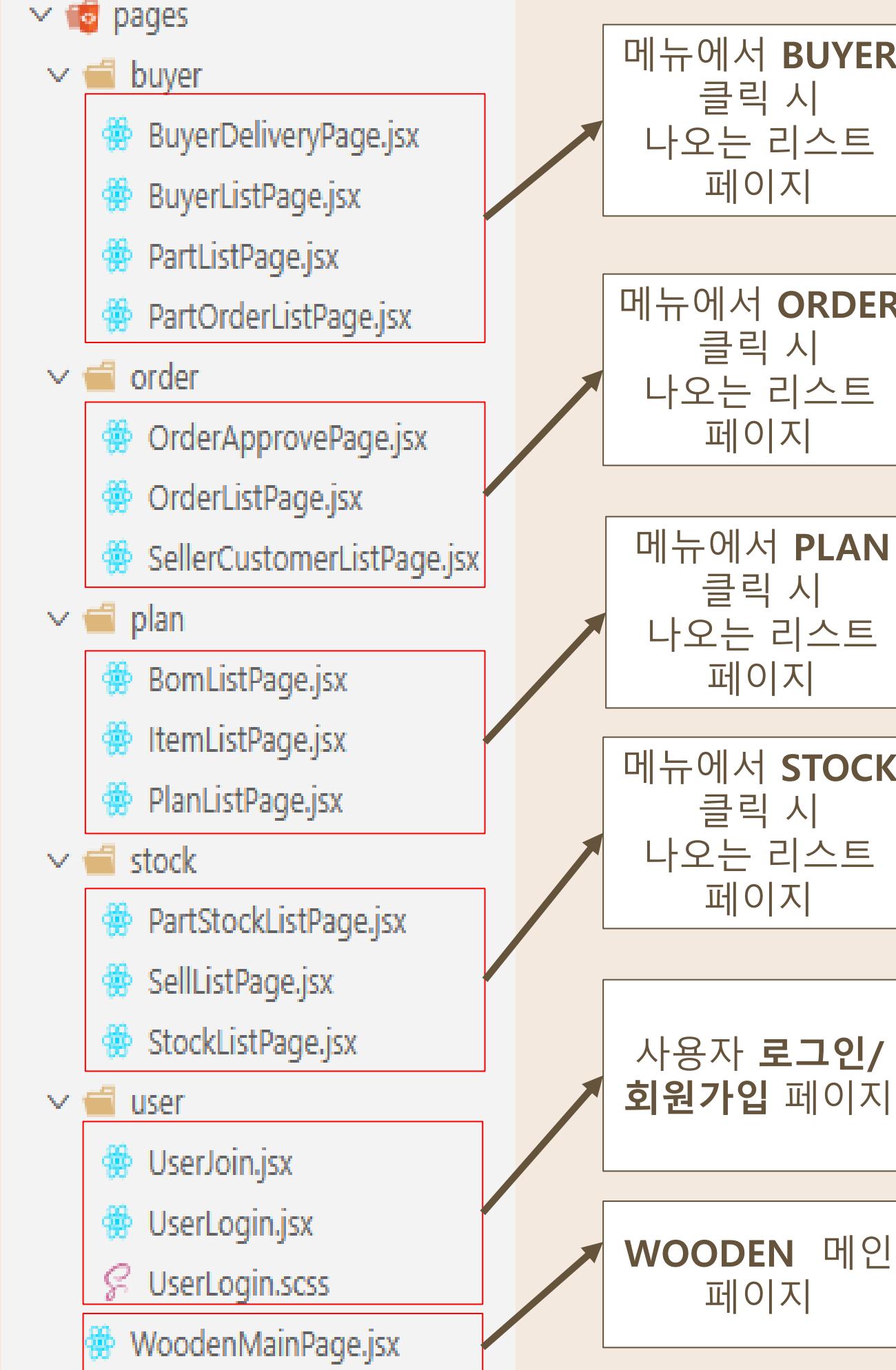
페이지들마다  
테이블, 모달 컴포넌트를 통해  
나오는 form의 내용을 정리한 배열



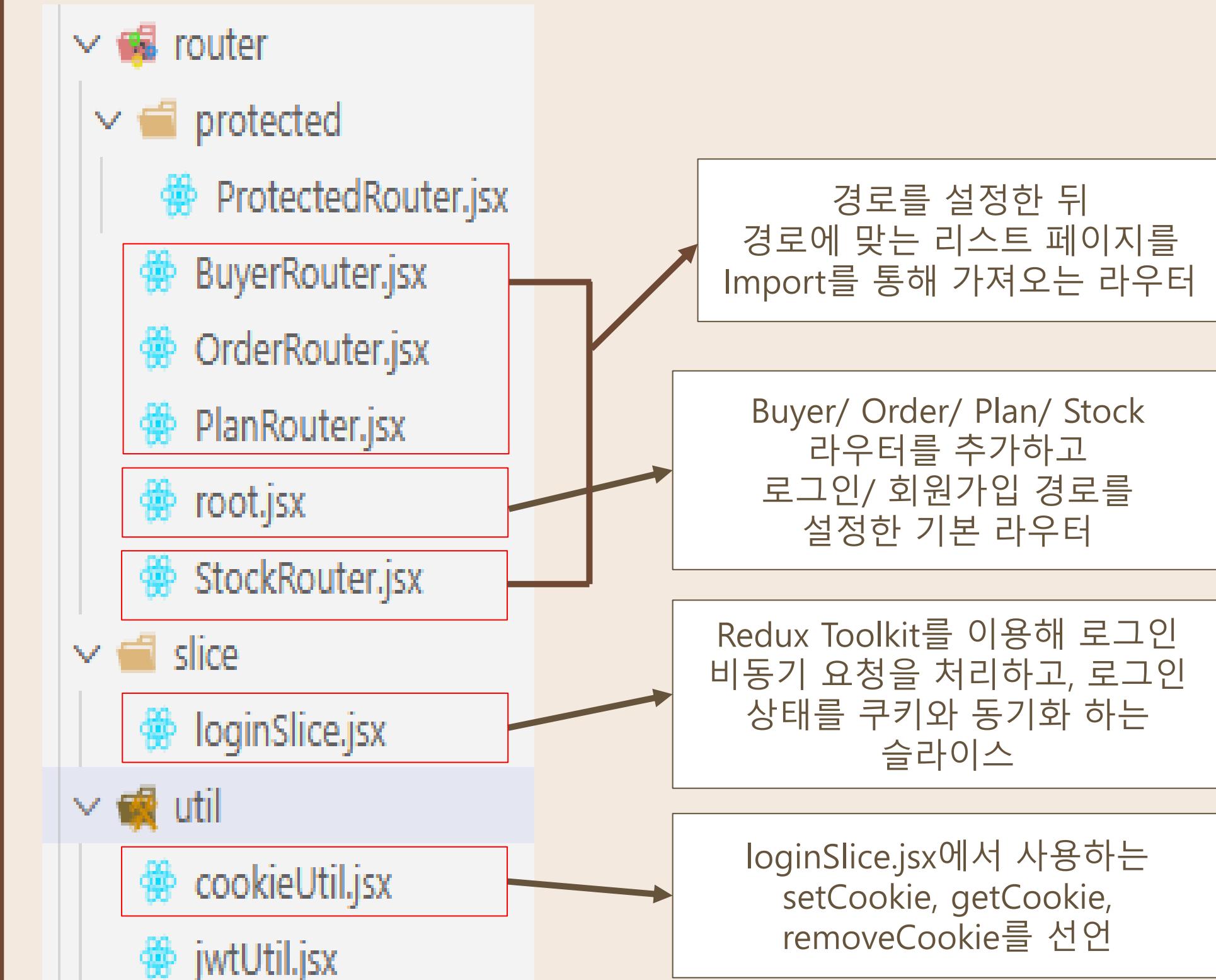
페이지 전체에서 각각 사용되는 컴포넌트



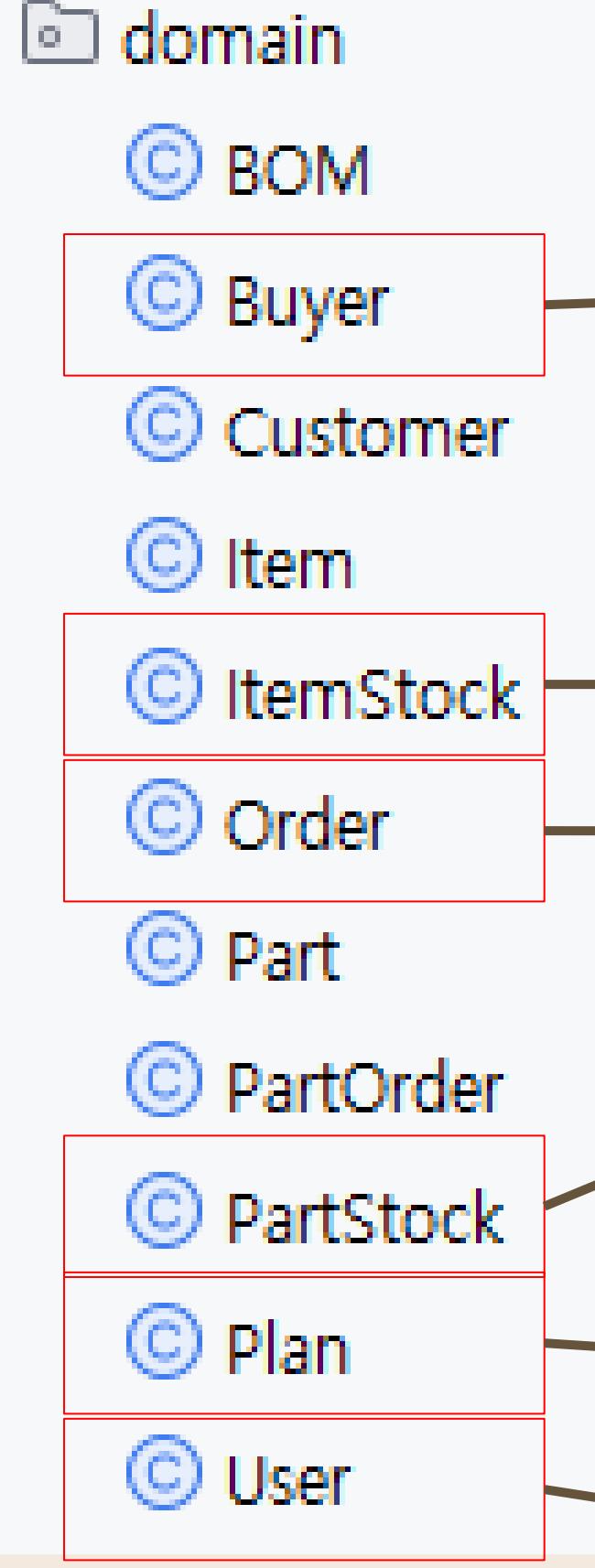
# Front-end 구조



loginSlice의 isLogin의 상태값에 따라, 사용자가 로그인 여부를 검사해 False면 “/login”으로 리다이렉트하거나, true면 “/”으로 이동 시키는 기능을 담당



# Back-end 구조



모든 DTO는 RequestDTO(클라이언트->서버  
요청 DTO)와  
ResponseDTO(서버->클라이언트 응답 DTO)로  
분리해서 작성



# 공통 흐름

Client : **useCRUD로 CRUD 표준화** → ModalComponent/SearchComponent/InlineSelectCell로 UI 일관성 확보. 디바운스 검색, 낙관적 상태 전환, 버블링 차단이 기본 규칙.  
Server : 도메인별 REST만 제공하면 됨. 공통 모듈은 엔드포인트를 소유하지 않고 api 주입 규약으로 호출·검증·에러 메시지를 재사용.

## FrontEnd 공통 [훅 / 컴포넌트]

useCRUD.jsx

**핵심 동작** : getAll()로 목록 로드 → items 세팅 → openCreate() → formData 초기화, 등록 모달 오픈 → openEdit(row) → 선택행을 selectedItem/formData에 주입, 수정 모달 오픈 → handleCreate/Update/Delete() → 주입된 api 호출, 성공 시 true 반환 → 각 페이지에서 성공 후 재조회(refetch) 또는 낙관적 갱신

**비고** : useCRUD 내부에 정의된 const [items, setItems] = useState([]); 만 페이지마다 호출해 재정의 후 사용함

ModalComponent.jsx

**Props** : isOpen, onClose, onConfirm, title, children

**패턴** : 등록/수정/삭제 폼을 children으로 받음 바깥 클릭/닫기 버튼 → onClose 상단 타이틀 + 본문 + 확인 영역 레이아웃 고정

ButtonComponent.jsx

**props**: text, onClick, cln

**용도**: 새로고침/등록/수정/삭제 등 액션 버튼 스타일 일관화

BackButtonComponent.jsx

**props**: text만 주입, 모든 페이지에서 동일 UX

**동작**: navigate(-1) 호출

SearchComponent.jsx

**props**: value, onChange, onDebounced, delay, minLength, placeholder, className

**흐름**: 즉시 값은 onChange로, 디바운스된 값만 onDebounced로 전달 → 각 페이지는 term으로만 필터(useMemo) 수행

**효과**: 타이핑 중 불필요한 필터/요청 억제

**props**: value, onChange, onDebounced, delay, minLength, placeholder, className

**흐름**: 즉시 값은 onChange로, 디바운스된 값만 onDebounced로 전달 → 각 페이지는 term으로만 필터(useMemo) 수행

**효과**: 타이핑 중 불필요한 필터/요청 억제

InlineSelectCell.jsx

## BackEnd 주요 기능

### 주요 흐름

Controller → Service → Repository → Entity + Request/Response DTO

### ---API 주입 규약---

- getAll() → any[]: 목록 조회
- create(fd) → any: 등록
- update(fd) → any: 수정 (id 포함 또는 경로 변수)
- delete(id) → void: 삭제
- 페이지별로 엔드포인트만 다름 호출·오류 처리 패턴은 동일.

### ---상태형 엔드포인트 패턴---

인라인 상태 변경 : ex ) PATCH /resource/{id} or /status  
낙관적 UI: 선택 즉시 로컬 변경 → 서버 실패 시 롤백  
검증 에러/비즈니스 에러: 서버가 409/400과 메시지 반환 →  
공통 alert 으로 노출

### ---재조회 전략---

등록/수정/삭제 완료: refetch()로 서버 소스 true 반영  
상태 변경: 성능/UX 고려해 낙관적 갱신 후, 필요 시 조건부  
재조회 ( ex : 완료 항목은 목록에서 숨김)

## FrontEnd 공통 [SCSS]

-- 임포트 순서 --  
 App.scss -> Button.scss -> Modal.scss ->  
 Responsive.scss

기본 → 컴포넌트 → 반응형 덮어쓰기로 스타일  
**충돌 디버깅 방지**

App.scss

Button.scss

Modal.scss

Responsive1500.scss

**역할:** 글로벌 베이스 스타일. 리셋, 폰트, 바디/컨테이너, 헤더·로고·그리드 같은 전역 레이아웃.

**주요 포인트:**

\*{ margin:0; padding:0; } 류의 리셋과 폰트 기본값 지정.  
 헤더 영역, .mainLogo 최대 너비 등 사이트 공통 프레임 정의.  
 오버플로우·스크롤·z-index 기본 정책이 여기서 결정.

**역할:** 버튼 컴포넌트 스타일 모음. 공통 버튼, 백버튼, 아이콘 버튼, 호버/활성 상태 등.

**주요 포인트:**

.backbtn-wrapper 래퍼와 내부 ::before/::after 같은 아이콘 라인 구현 흔적.  
 상태 클래스 조합: .btn, .btn--primary, .btn--ghost 식으로 확장하는 구조가 적합

**역할:** 모달 레이어. 오버레이, 모달 컨테이너, 열림/닫힘 트랜지션, 스크롤 잠금 처리의 근간.

**주요 포인트:**

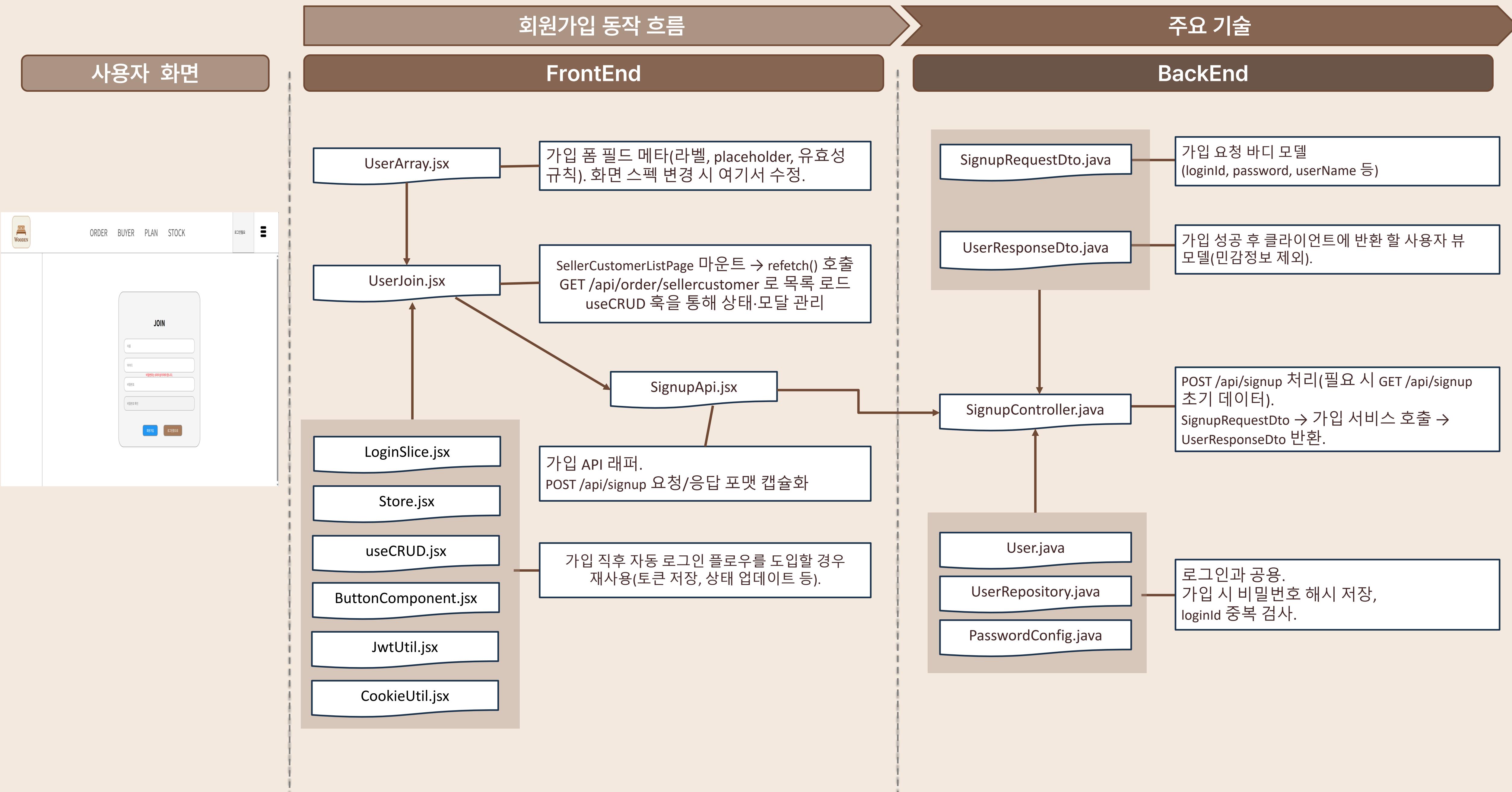
.modal 루트, .modal\_\_overlay 전면 덮개, position: fixed; top:0; left:0; z-index 고정.  
 페이드/스케일 애니메이션으로 등장/퇴장. opacity: 0 → 1, transform 활용.  
 .is-open 같은 상태 클래스 기반으로 표시 제어.

**역할:** 반응형 브레이크포인트( $\approx\leq 1500px$ ) 전용 오버라이드. 레이아웃, 네비, 햄버거 메뉴 크기·정렬 같은 것들 축소·재배치.

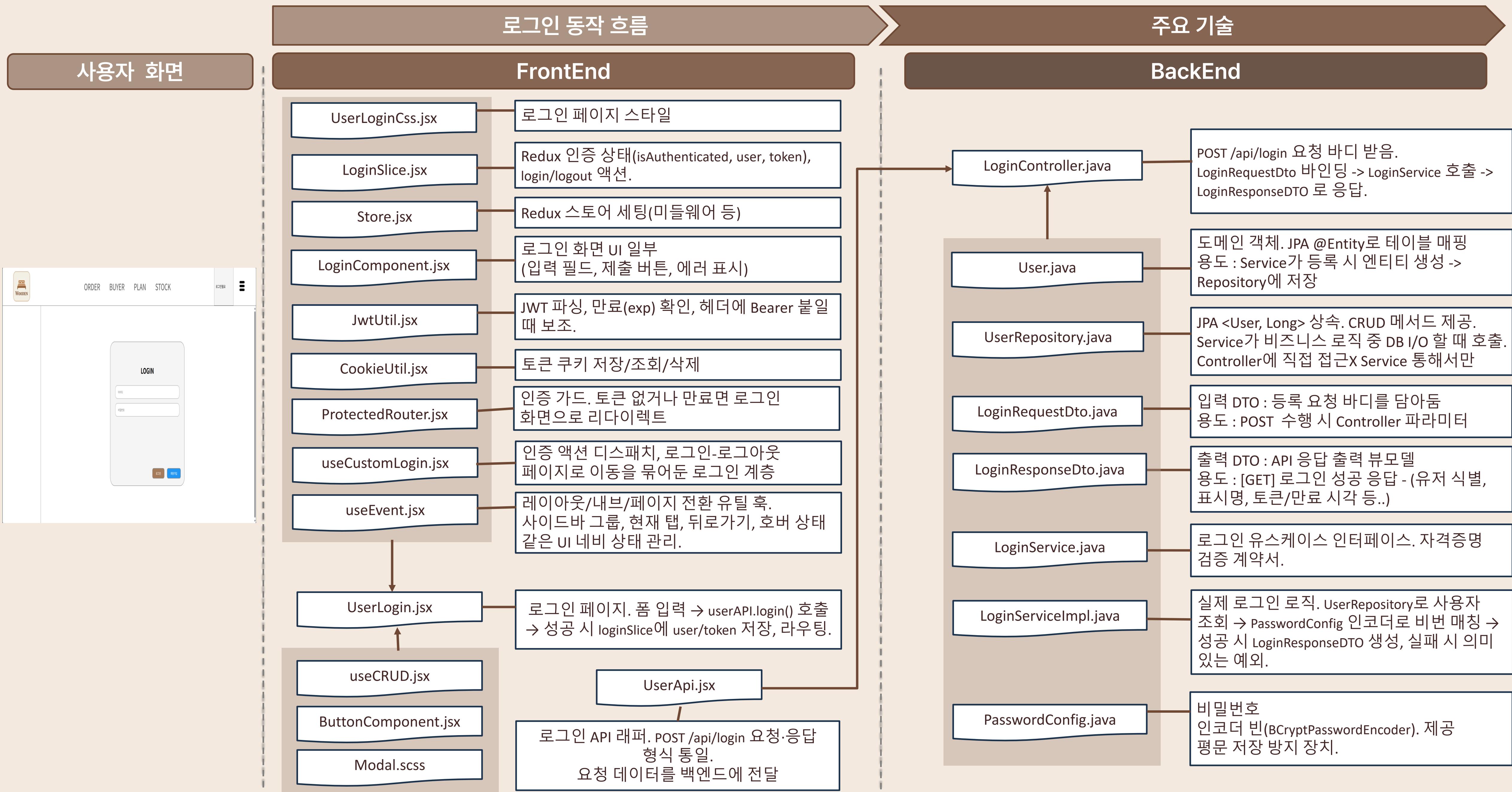
**주요 포인트 :**

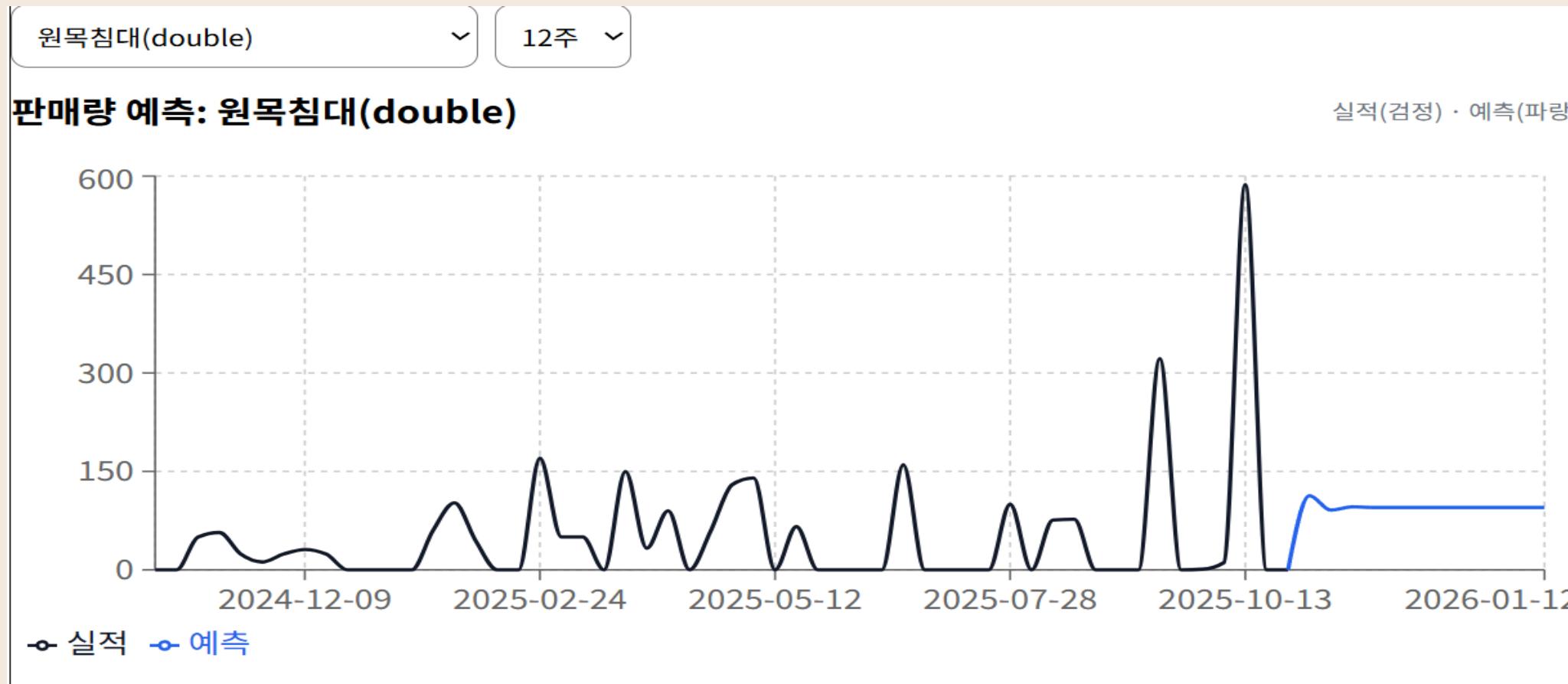
@media screen and (max-width: 1500px) 블록 안에서 헤더/그리드/사이드 UI 크기 조정. 햄버거 .hamburger 폭 비율 조정 같은 모바일 전환 전 단계 튜닝.  
 공통 스타일(App.scss)보다 우선 적용되도록 마지막에 임포트하는 게 정석

# UserJoin

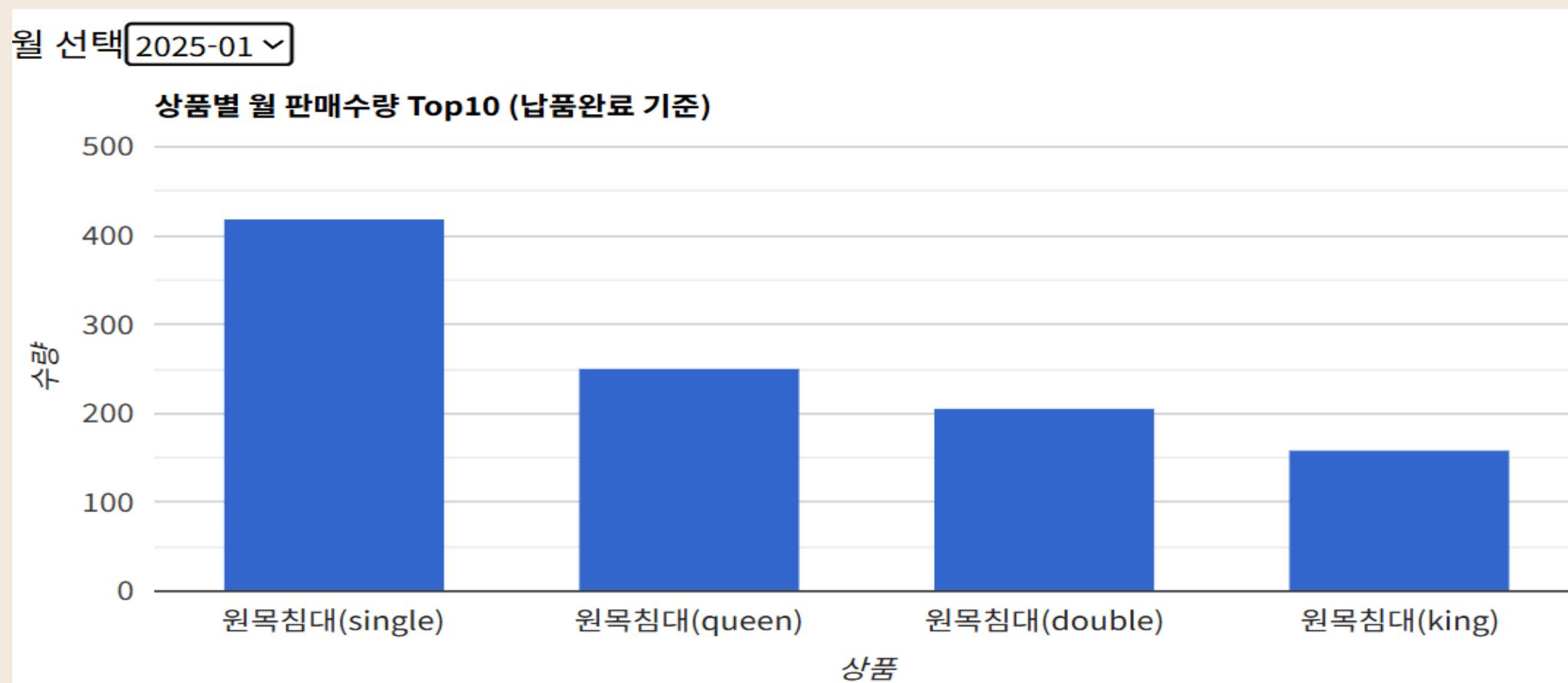
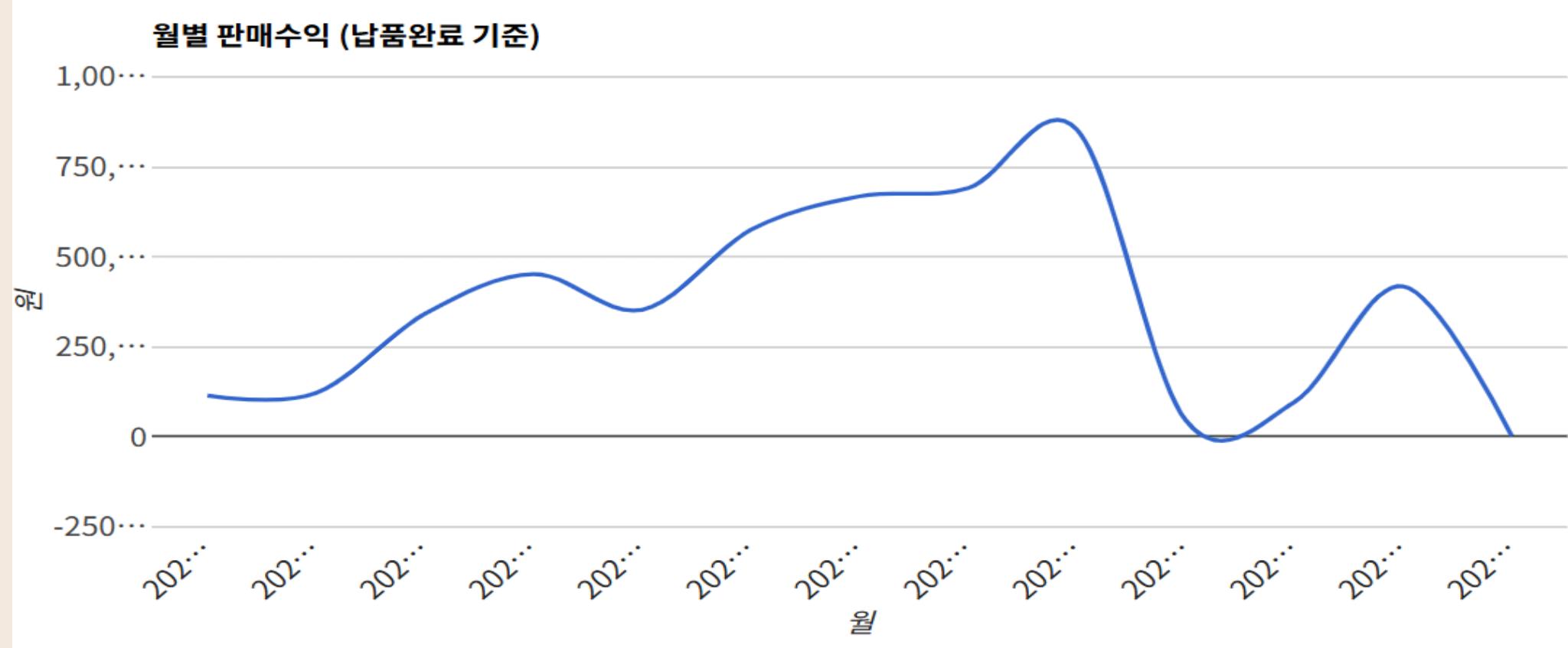


# UserLogin



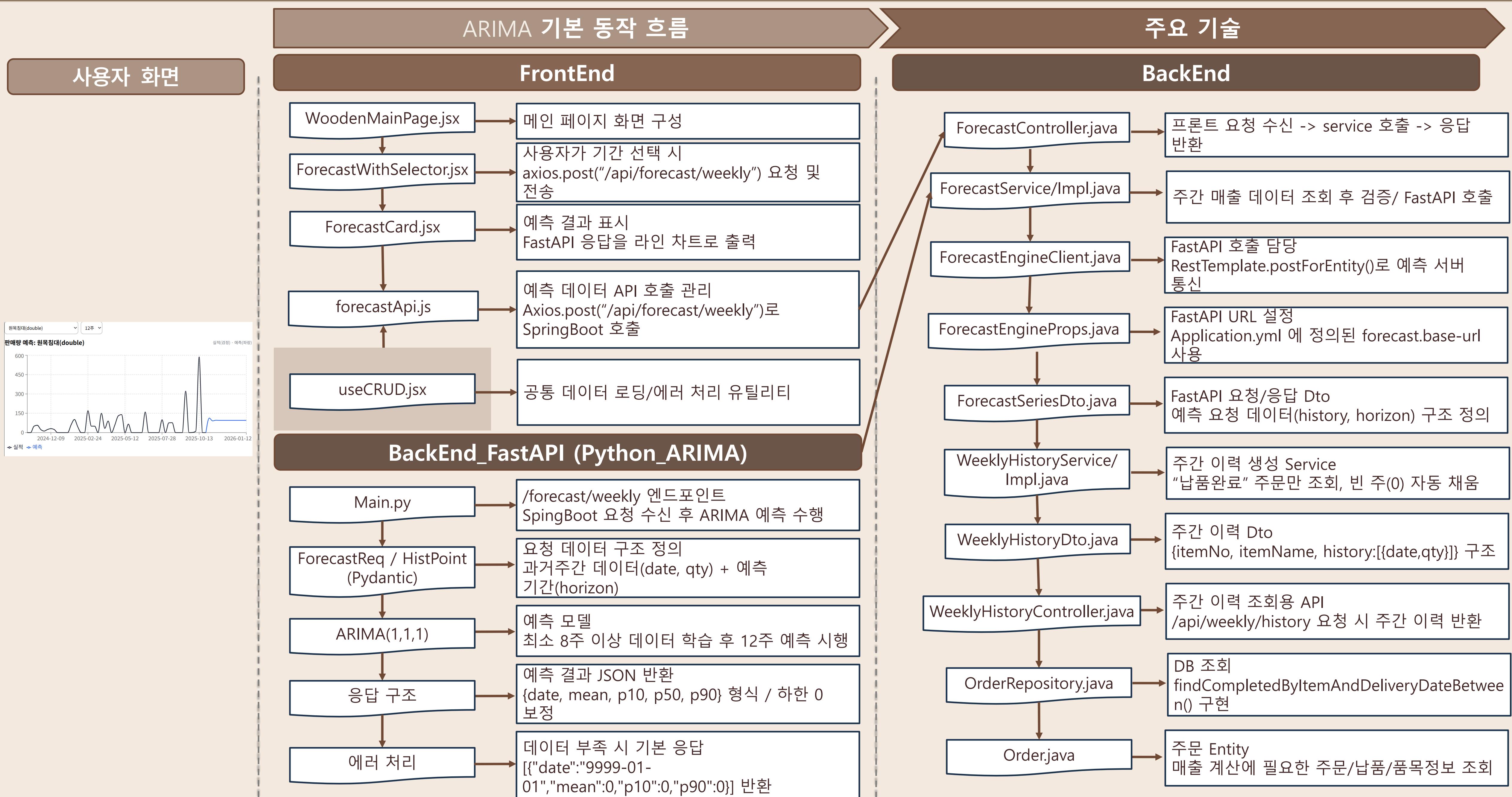


1. ARIMA 예측은 pandas/numpy로 주차 별 보정을 한 뒤, statsmodels의 **ARIMA**로 학습·예측을 수행
2. React + Recharts 라이브러리를 이용하여 화면에 표시, matplotlib으로 시각화



1. 백엔드에서 승인완료+납품완료 주문만 집계해 월별 합계와 해당월 품목 Top10 수량을 전달한다.  
프론트는 **Google Charts**로 월별 매출과 선택월 Top10을 렌더링하며, 상단 드롭다운으로 월을 바꾸면 차트가 갱신된다.

# 1 – 1 . Main Page (주간별 판매량 예측 ARIMA Chart)



# 1 – 2 . Main Page (월별 매출액/월별 상품별 판매수량)

Chart)

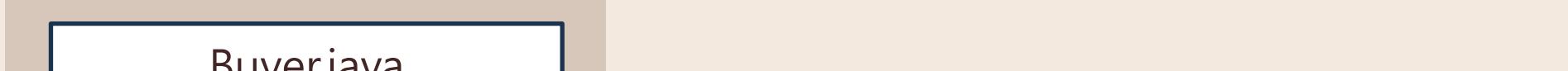
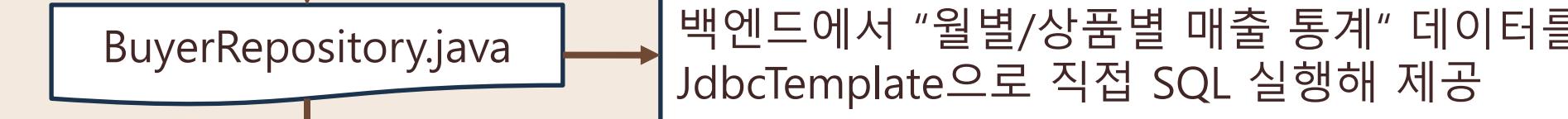
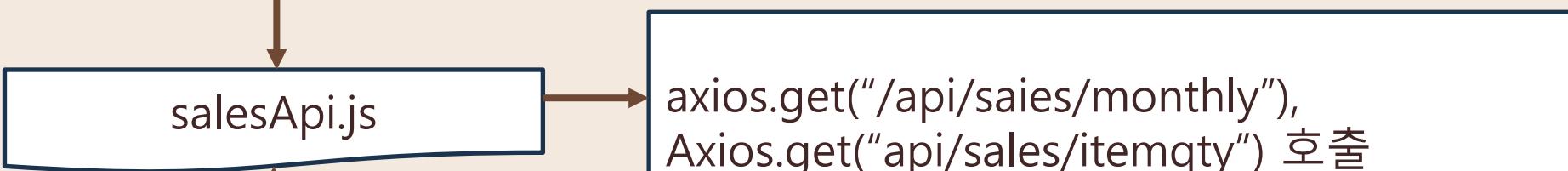
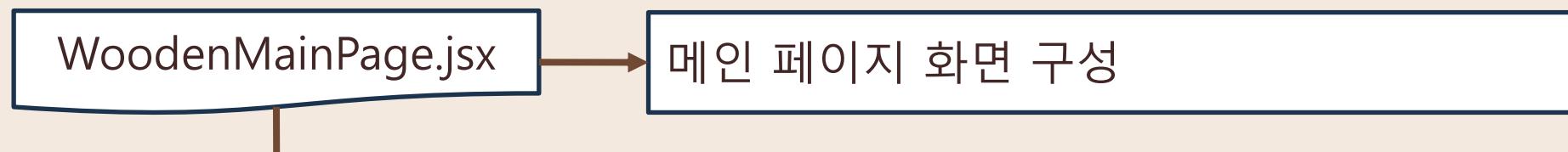
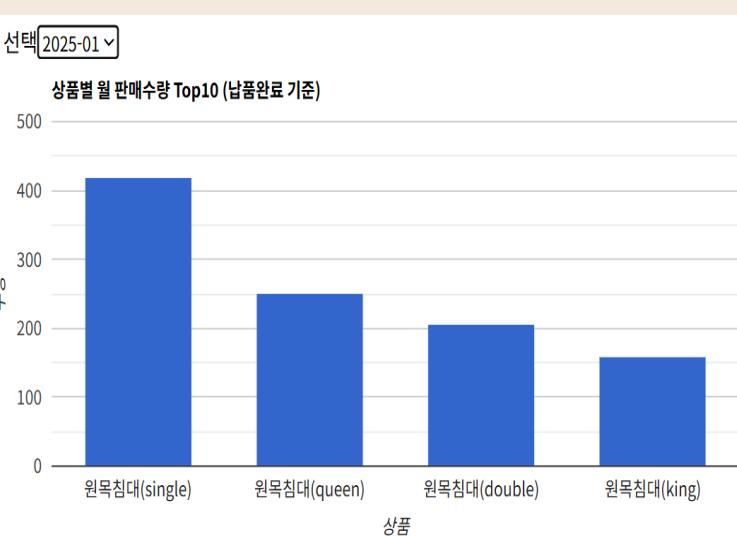
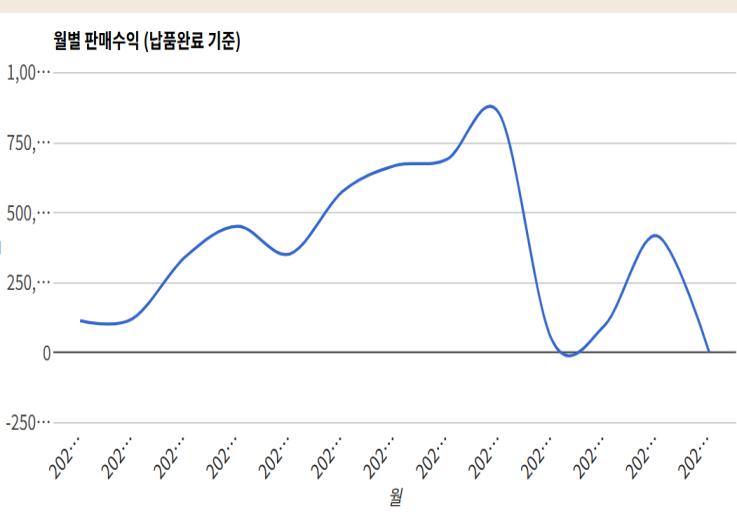
사용자 화면

매출 / 상품 별 차트 기본 동작 흐름

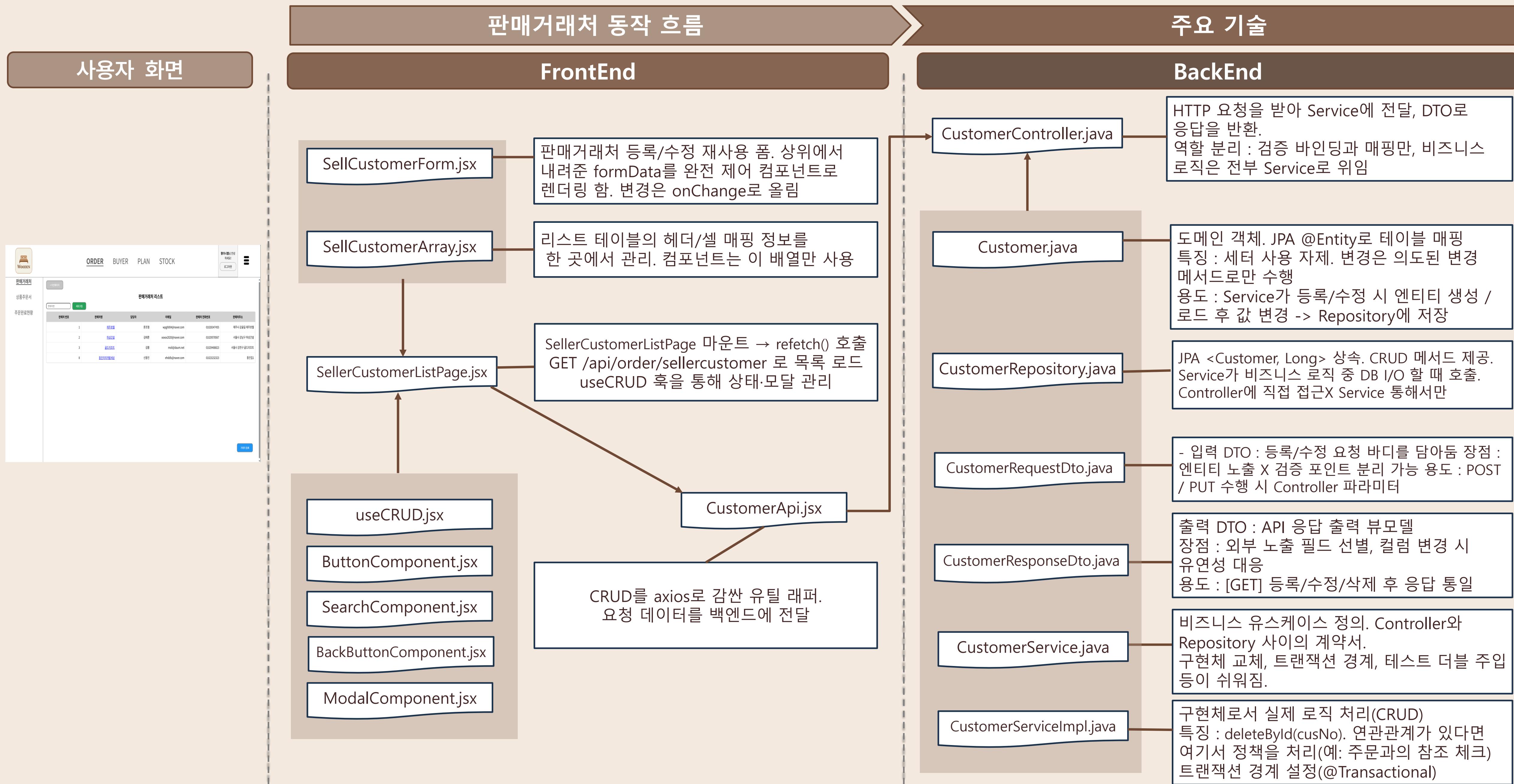
주요 기술

FrontEnd

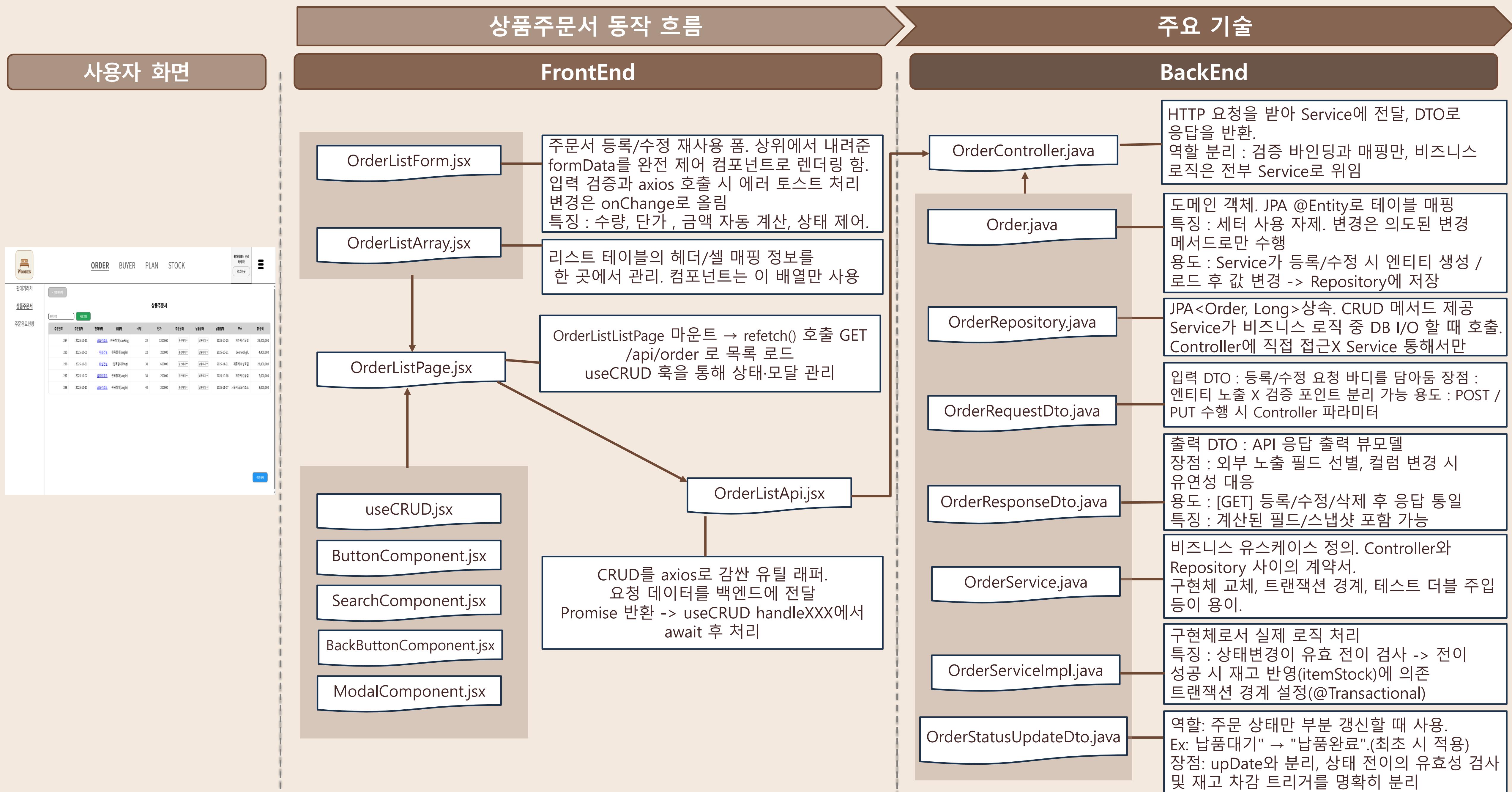
BackEnd



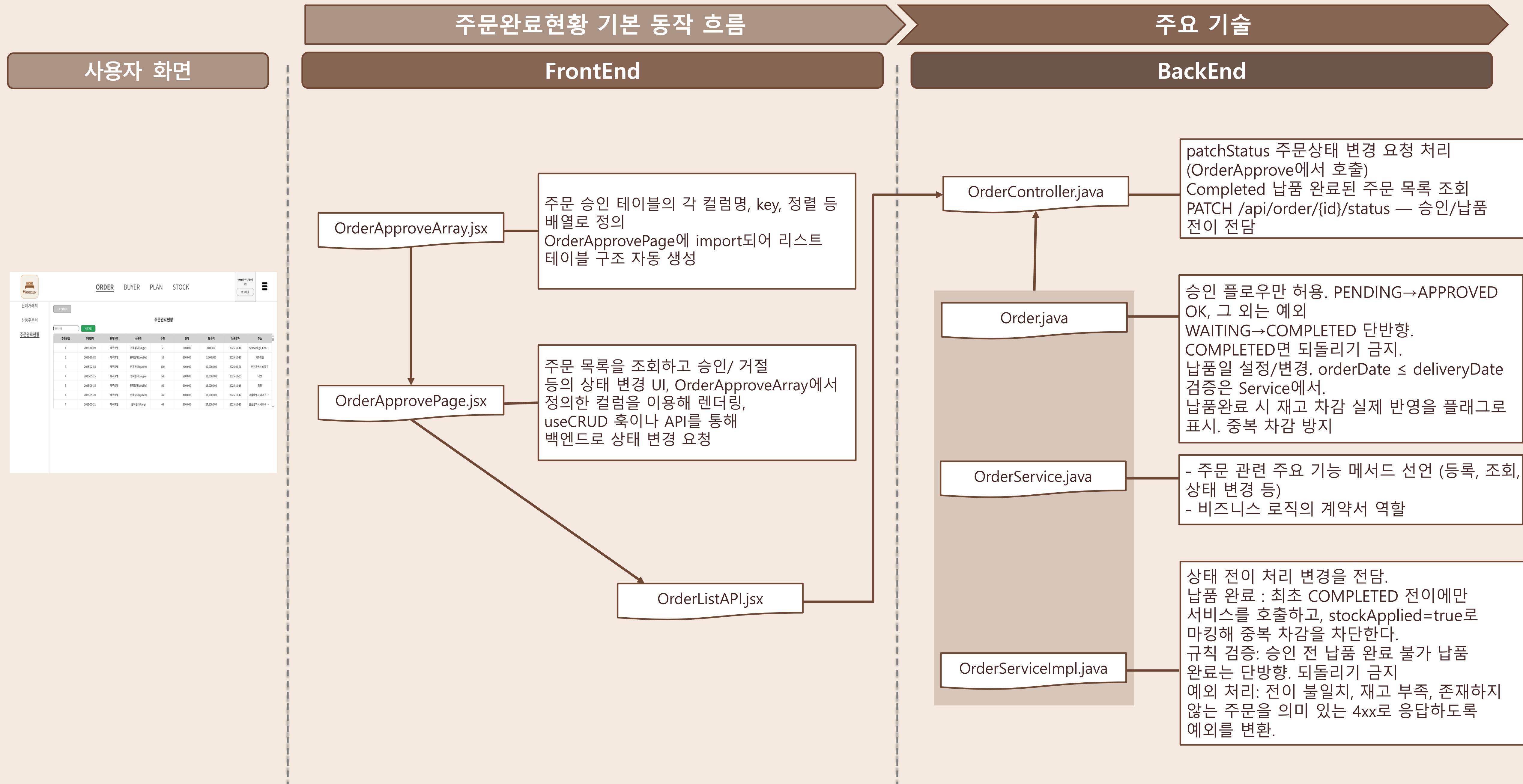
# ORDER . 판매거래처



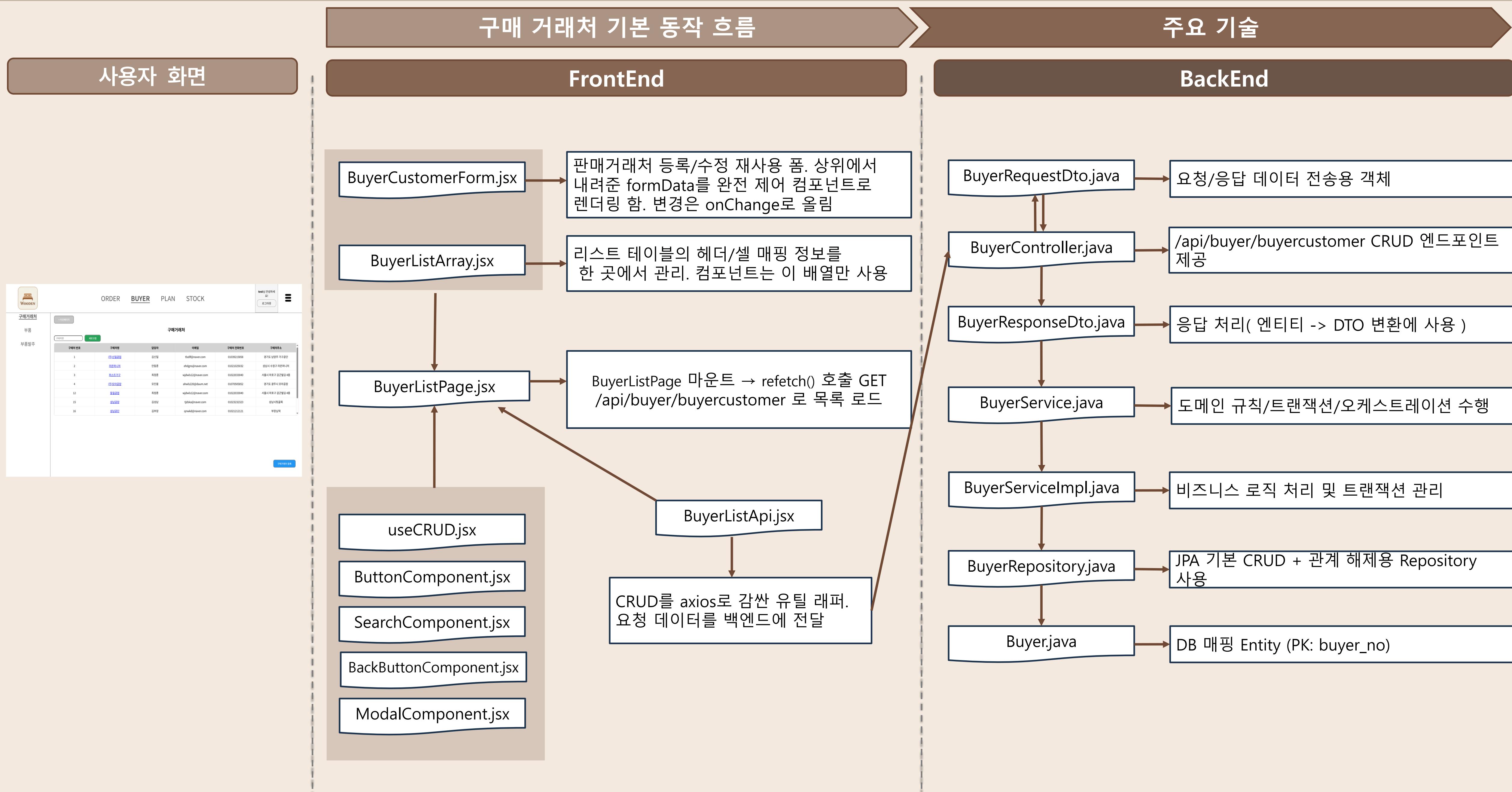
# ORDER . 상품주문서



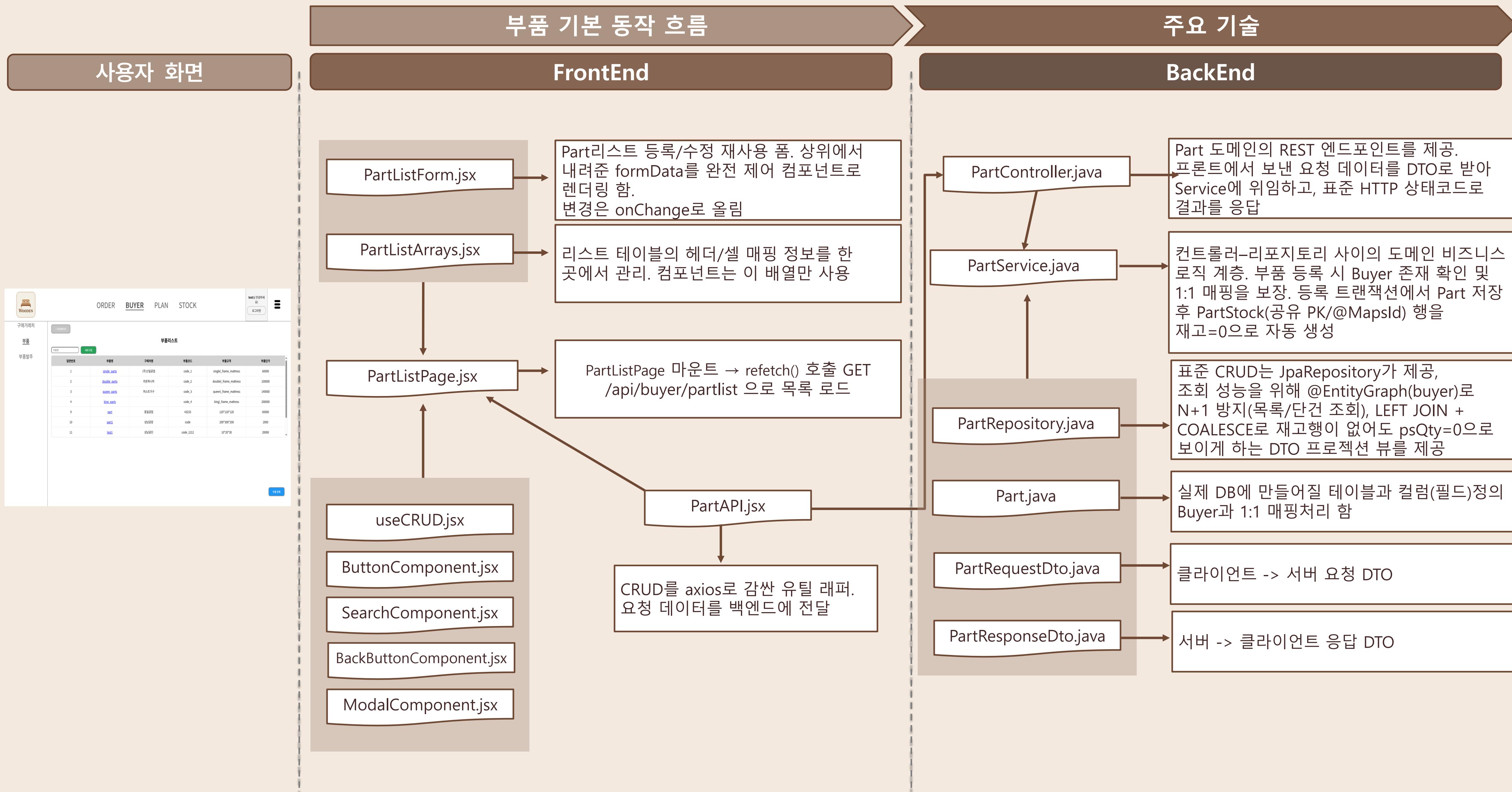
# ORDER . 주문완료현황



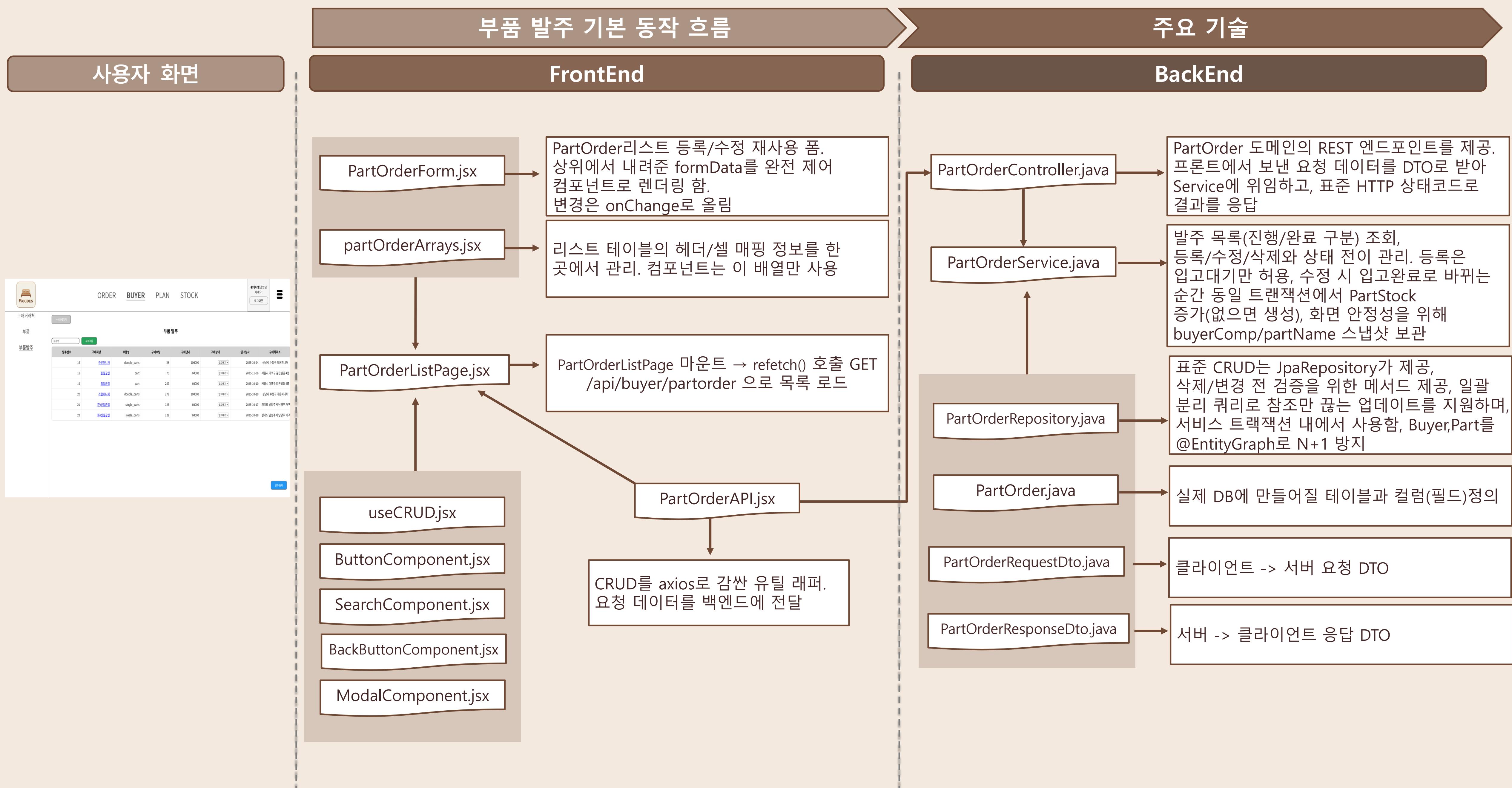
# BUYER . 구매 거래처



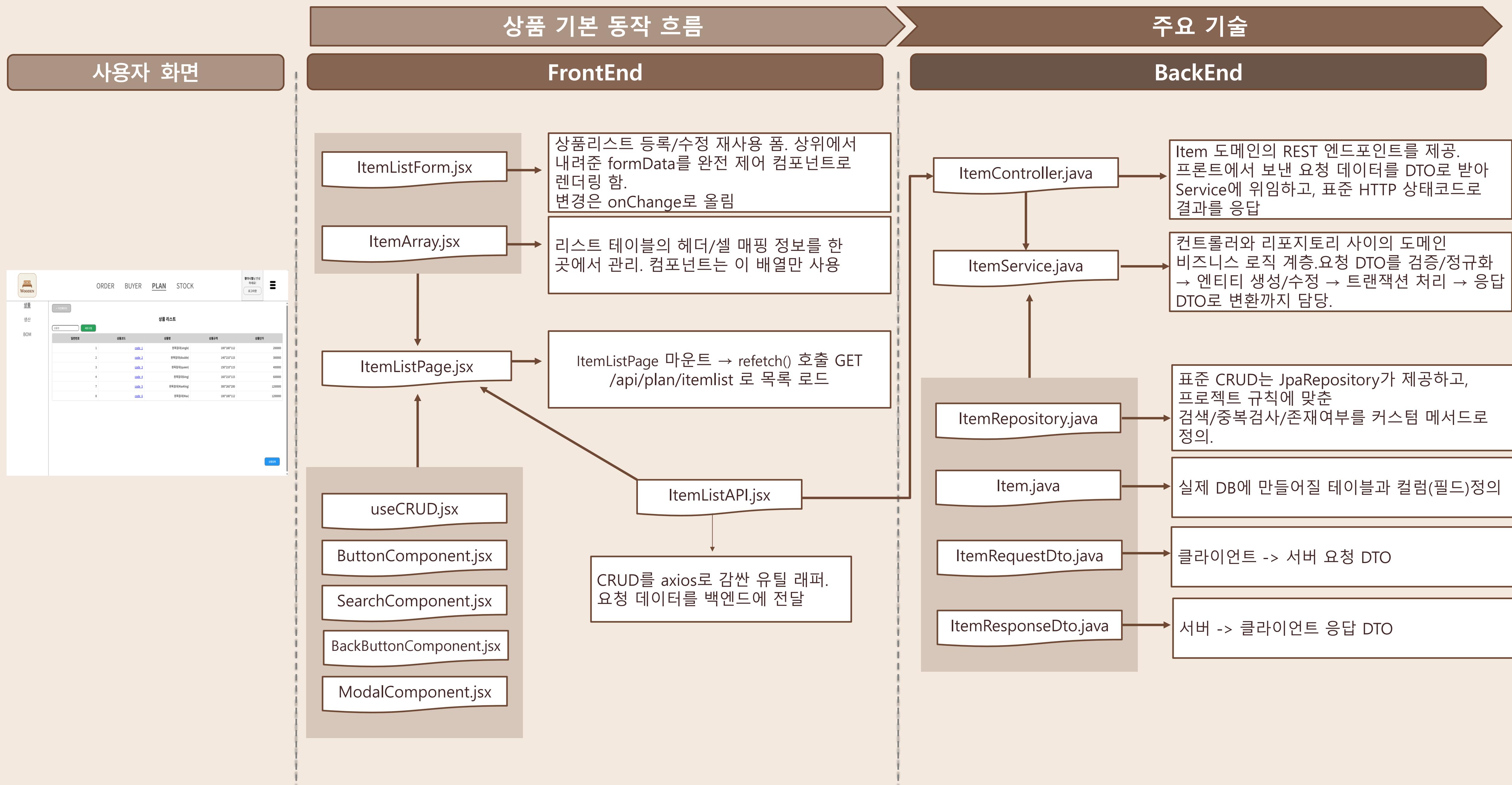
# BUYER . 부품



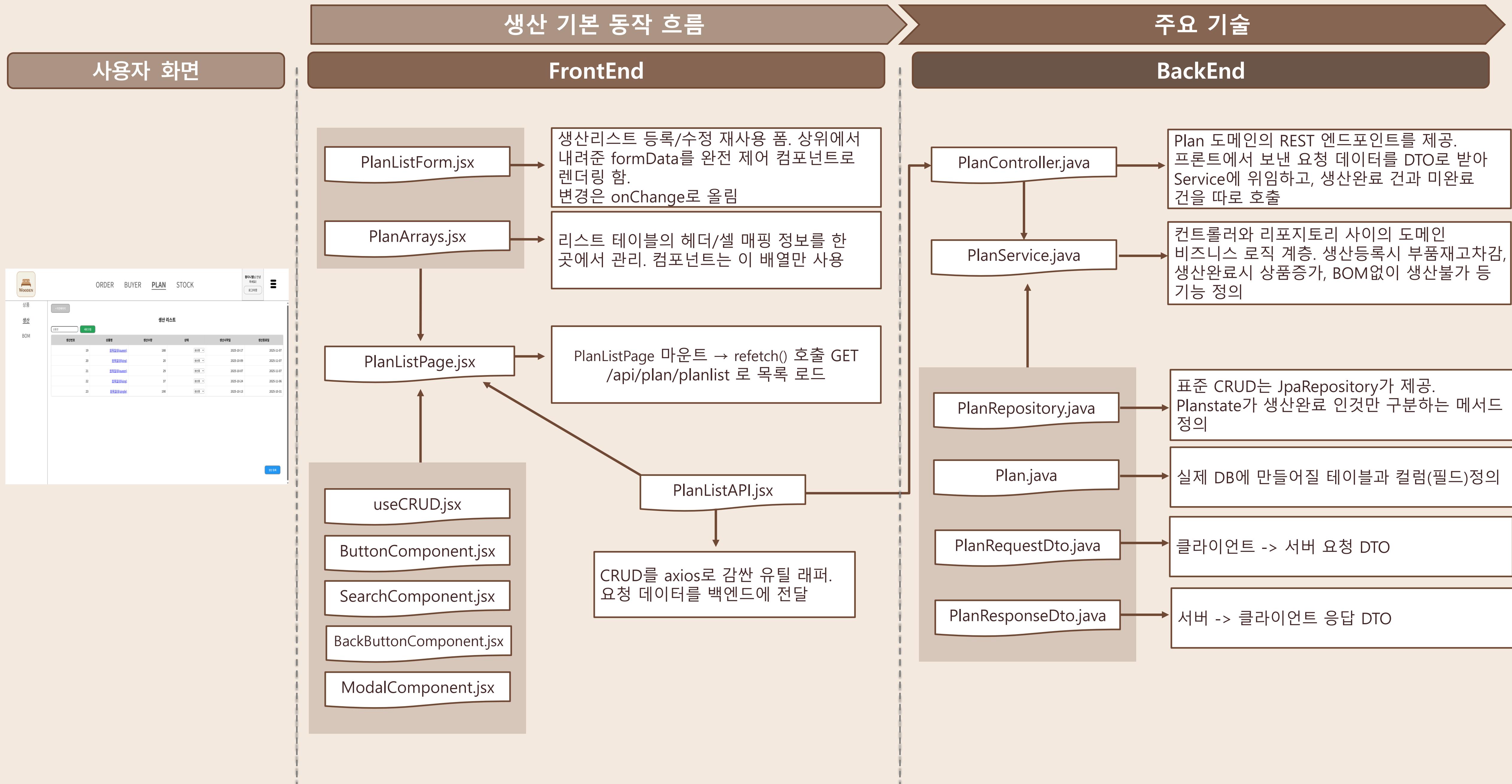
# BUYER . 부품 발주



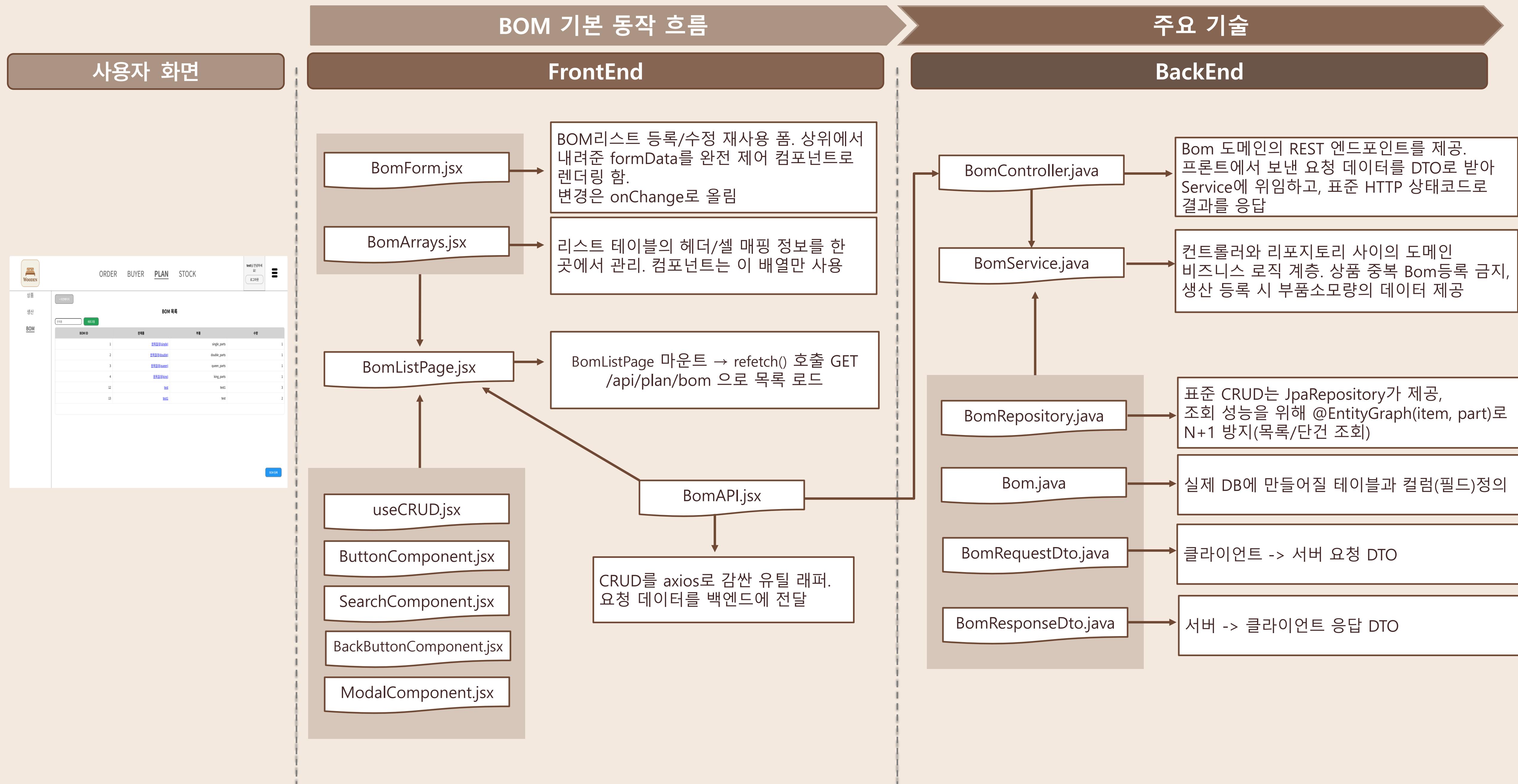
# PLAN . 상품



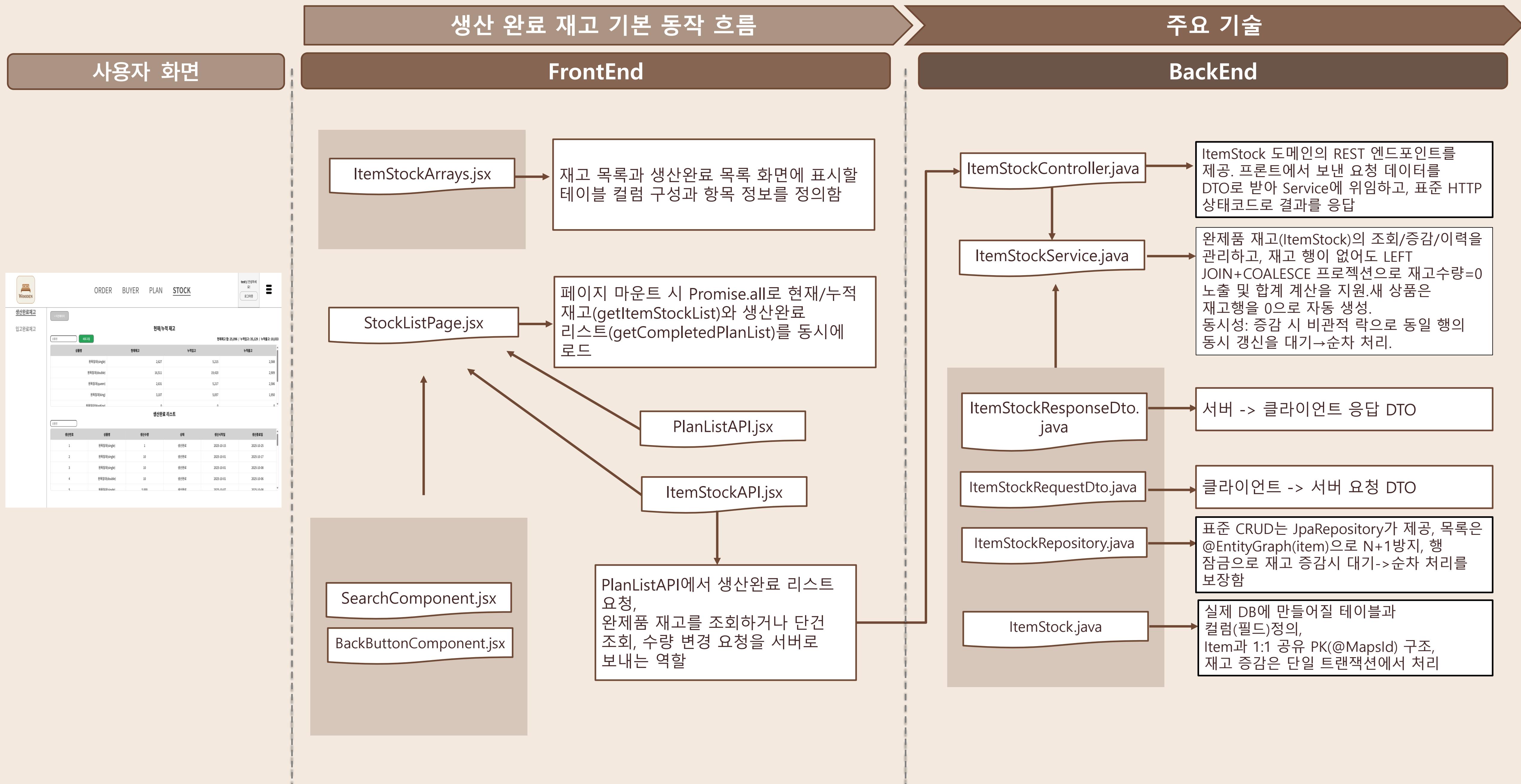
# PLAN . 생산



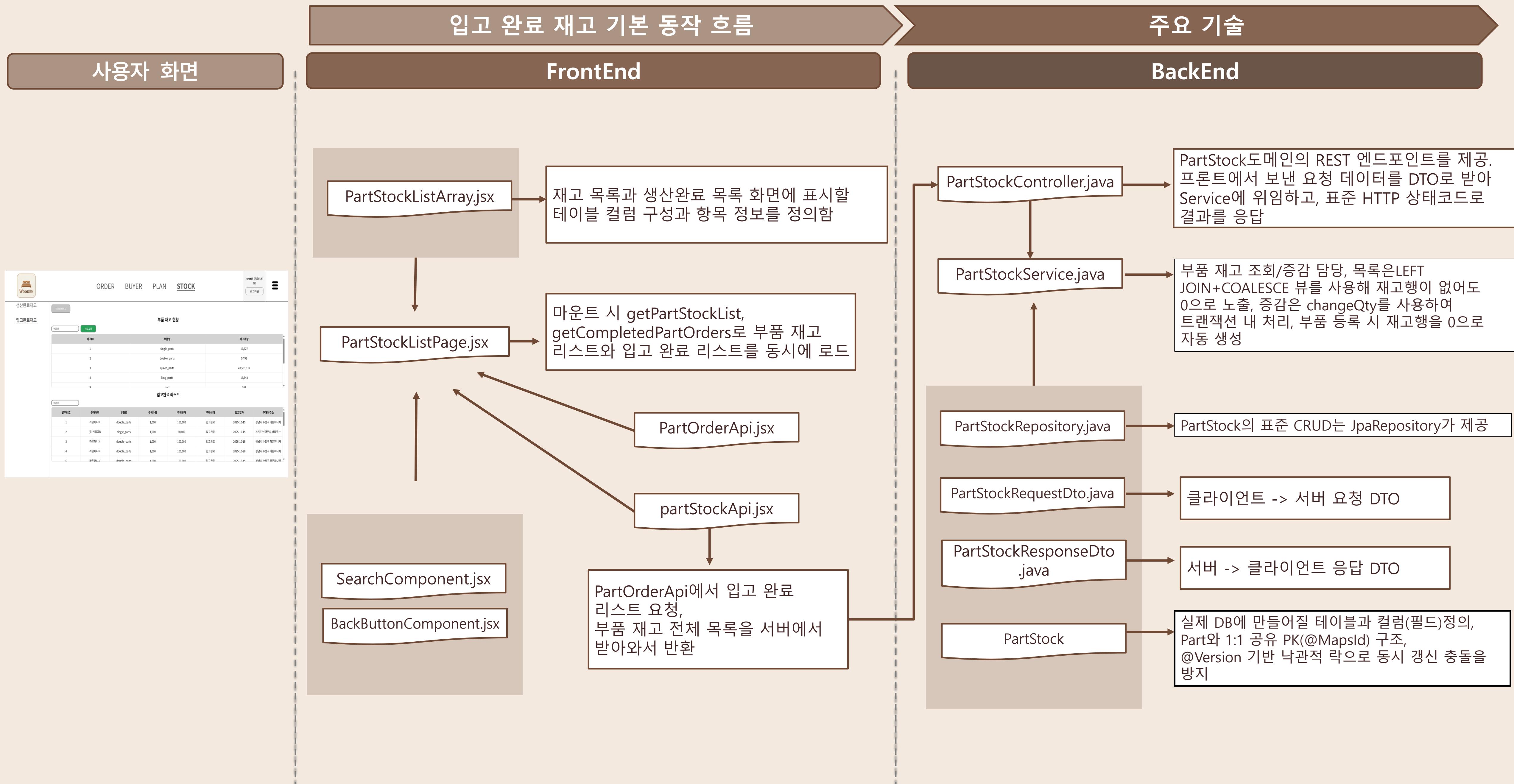
# PLAN . BOM



# STOCK . 생산 완료 재고



# STOCK . 입고 완료 재고



# - 후기 |

황 다니엘

WOODEN 프로젝트를 시작하는 과정에서 서로의 생각이 다르기 때문에 의견 회의도 잣았고 많이 서툴렀지만 대화가 많았던 만큼 이후엔 원활히 진행하게 됐습니다. 모두가 프로젝트에 몰두해 있을 시점엔 팀원들 각자가 잘 이해하는 부분과 그렇지 못한 부분을 서로 돋고 채워 가다 보니 더욱 가까워질 수 있었고 저 또한 많은 공부가 되었습니다. 프로젝트 완성이라는 공통된 목적 안에서 제가 생각하지 못했던 방법들이 오고 가는 것에도 많은 재미를 느꼈고 교육 기간 동안 깊은 고민들과 어려움이 많았지만 배운 내용들을 활용하고 응용하면서 개발에 몰입하고 WOODEN 을 다 같이 완성 시켰다는 것에 성취감이 컸습니다.

