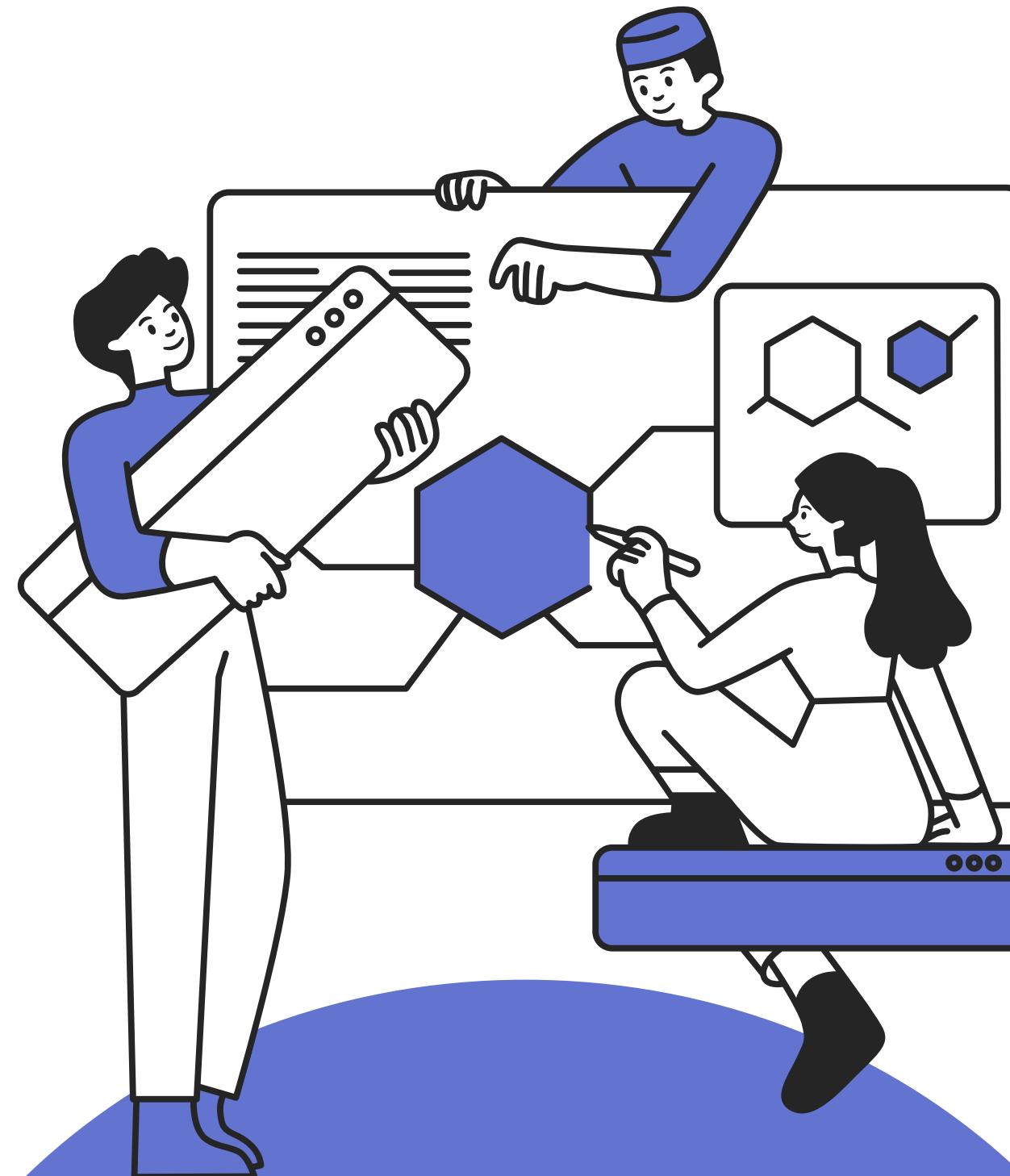


JAVA GUI와 JDBC 연동을 활용한 양방향 화물 배차 관리 시스템 구현

인공지능소프트웨어학과 1학년

2501110200 김재영



CONTENTS

- 1 주제, 주제선정이유
- 2 E-R 다이어그램
- 3 홈화면 및 로그인/회원가입
- 4 고객 홈화면
- 5 고객-배송신청(Insert)
- 6 고객-주문내역(Search)
- 7 고객-내정보수정 및 주문수정/취소
(Update/Delete)
- 8 기사 로그인 및 홈화면
- 9 기사-실시간콜(Search)
- 10 기사-배차내역(Search)/배
차관리(Update/Delete)
- 11 기사-내정보수정(Update)

01

주제, 주제선정이유

주제

JAVA GUI와 JDBC 연동을 활용한 양방향 화물 배차 관리 시스템 구현 (프로젝트명: Pick & Go)

주제선정이유

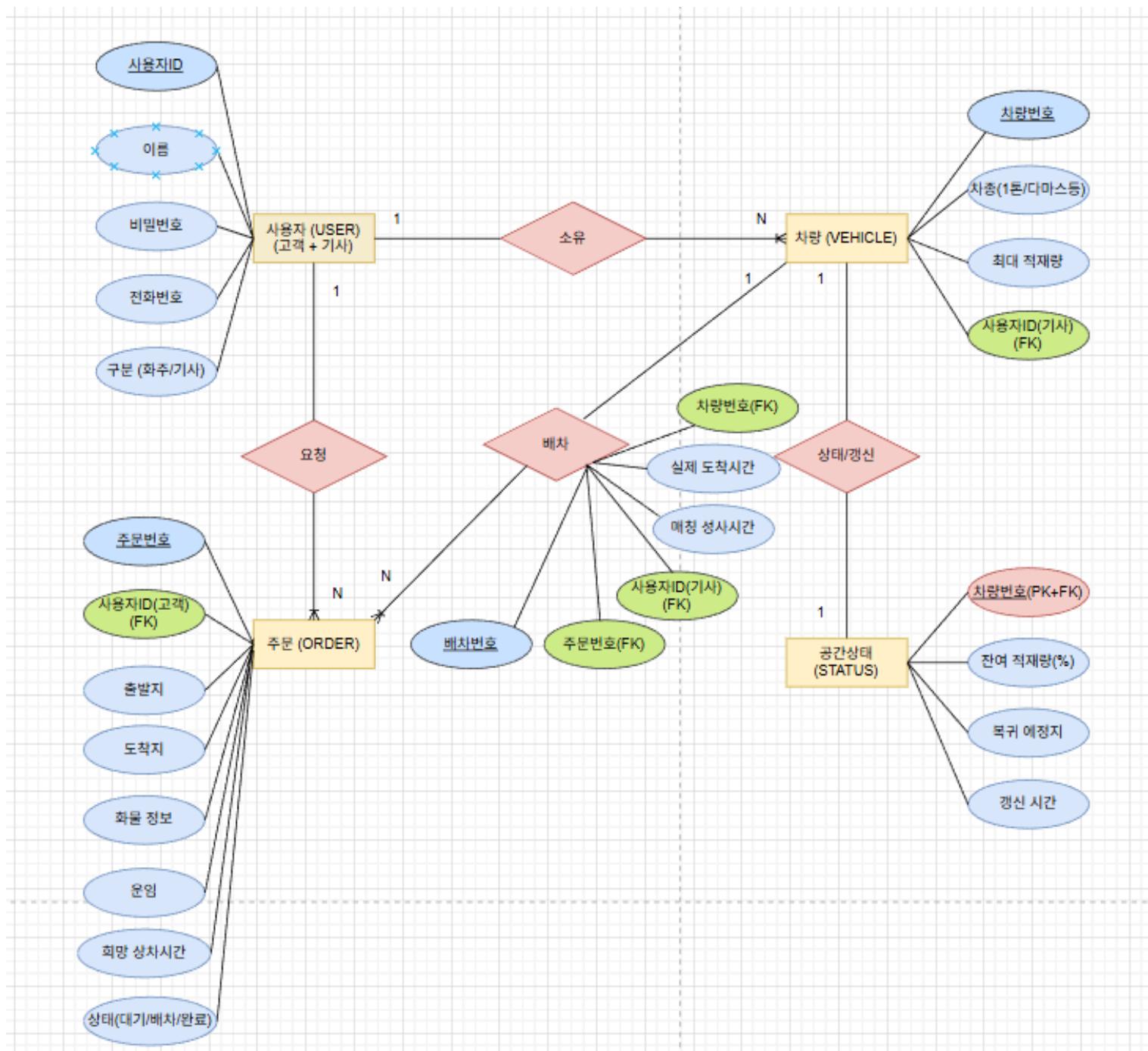
이번 학기 수업 시간에 배운 JDBC 프로그래밍과 GUI 설계를 가장 효과적으로 보여줄 수 있는 주제가 무엇일지 깊이 고민했습니다. 보통 예제로 많이 다루는 도서 관리나 학사 관리 시스템보다는, 실생활에서 실제로 쓰이는 '플랫폼' 형태의 프로그램을 직접 구현해보고 싶다는 욕심이 생겼습니다.

최근 물류 시장이 커지면서 화주(고객)와 차주(기사)를 연결해 주는 배차 앱이 필수적이라는 점에 착안하여, 고객은 간편하게 화물을 등록하고 기사는 실시간으로 이를 확인해 배차받는 '화물 배차 시스템'을 주제로 선정하게 되었습니다.

특히 이 주제는 '고객'과 '기사'라는 서로 다른 권한을 가진 사용자가 하나의 데이터베이스를 공유하며 실시간으로 상호작용해야 하는 구조입니다. 따라서 수업 시간에 강조 하셨던 데이터베이스의 정규화, 트랜잭션 처리, 그리고 CRUD 기능을 종합적으로 학습하고 적용해보기에 최적의 주제라고 판단하여 선정했습니다.

End

E-R다이어그램



네, 알겠습니다! 보고서 줄글 형식으로 바로 복사해서 붙여넣으실 수 있도록 문단을 이어서 정리했습니다.
[E-R 다이어그램 및 데이터베이스 설계 의도]

본 프로젝트의 데이터베이스는 화주(고객)와 차주(기사)를 중개하는 화물 배차 플랫폼의 특성을 고려하여 데이터 무결성과 실무적인 확장성을 최우선으로 설계하였습니다. 우선 사용자(USER) 엔티티는 고객과 기사를 별도의 테이블로 나누지 않고 하나의 테이블로 통합 관리하되, user_type 컬럼에 CHECK ('고객', '기사') 제약 조건을 걸어 역할 구분을 명확히 했습니다. 또한 비밀번호는 향후 해싱 암호화를 고려해 VARCHAR2(255)로 넉넉하게 할당했고, 전화번호는 숫자형으로 저장 시 010의 앞자리 0이 소실되는 문제를 방지하기 위해 문자열(VARCHAR2) 타입을 사용했습니다.

차량(VEHICLE) 엔티티의 경우, 초기 기획 단계에서는 '1인 1차량'을 생각했으나 운수 회사 대표가 여러 대의 차량을 소유하고 기사를 고용하는 법인 및 임대 상황을 고려하여 한 명의 사용자가 여러 차량을 등록할 수 있는 1:N 구조로 확장성 있게 설계했습니다. 차종 또한 단순 모델명이 아닌 카고, 윙바디, 탑차 등 실제 물류 현장의 적재 형태를 기준으로 구분했습니다.

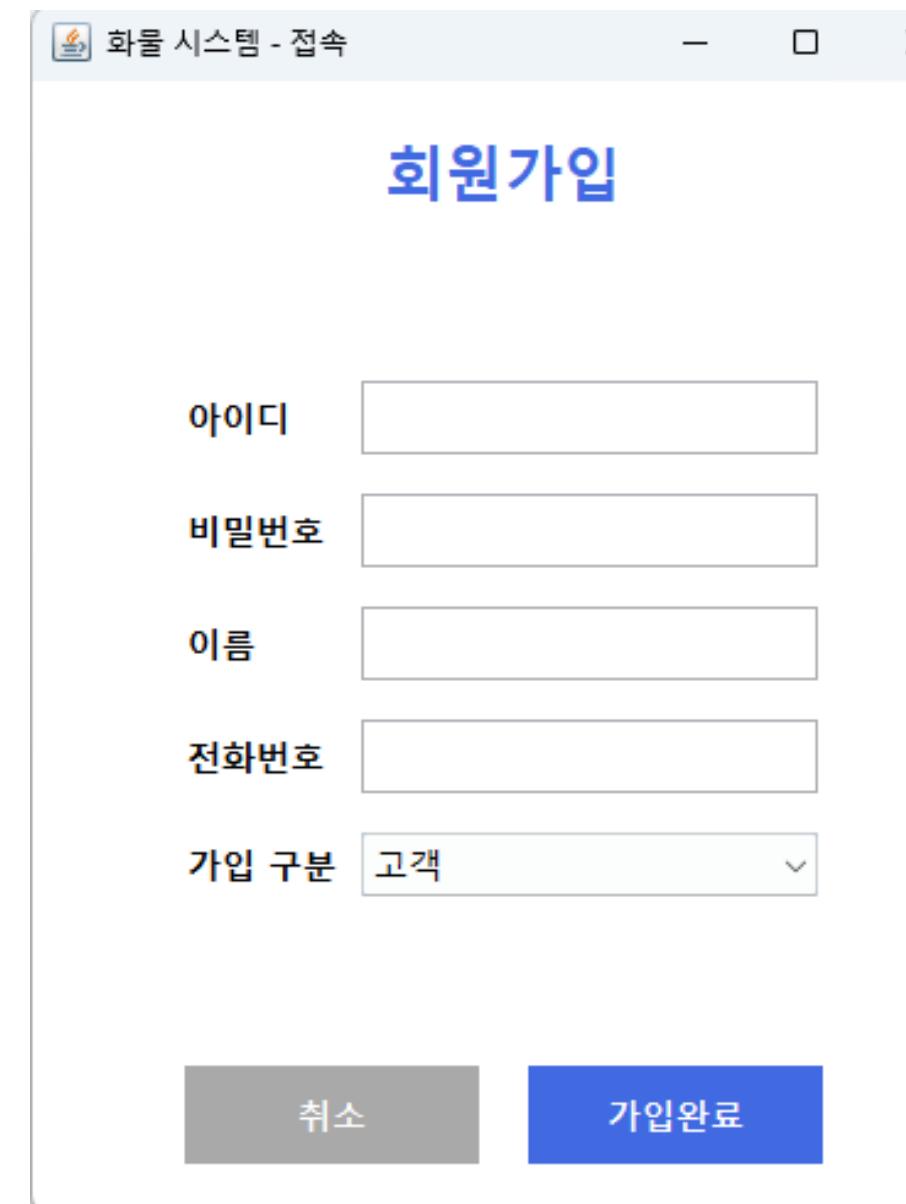
공간 상태(SPACE_STATUS) 엔티티는 차량의 고정 정보(차종, 최대 적재량)와 실시간으로 변하는 정보(현재 적재율, 복귀 예정지, 갱신 시간)를 분리하기 위해 설계했습니다. 관제 시스템 특성상 위치나 적재량 정보는 빈번하게 UPDATE가 발생하므로, 이를 정적인 차량 테이블과 분리함으로써 불필요한 락(Lock) 경합을 줄이고 조회 성능을 최적화했습니다. 이때 차량번호를 기본키(PK)이자 외래키(FK)로 설정하여 강력한 1:1 관계를 강제했습니다.

주문(ORDER_SHEET) 엔티티의 식별자는 단순 시퀀스 대신 YYYYMMDD-Seq 형태의 Formatted Key를 사용하여 날짜별 주문량을 직관적으로 파악하고 정렬 성능을 높였습니다. 동시 접속 시의 중복 채번 문제는 트랜잭션 내 FOR UPDATE 락을 통해 해결하는 로직을 전제로 했습니다. 또한 이론적으로 주문 상태는 배차 테이블의 존재 여부로 판단 가능하지만, 실무적으로 "대기 중인 주문만 조회하라"는 요청이 빈번하기 때문에 매번 JOIN을 수행하는 오버헤드를 줄이고자 주문 테이블에 status 컬럼을 두어 조회 성능을 극대화하는 역정규화를 적용했습니다.

마지막으로 배차(DISPATCH) 엔티티는 주문, 기사, 차량을 연결하는 핵심 매핑 테이블입니다. 앞서 설계한 차량 임대 시나리오에 맞춰 "정산을 받을 사람(기사)"과 "실제 운행된 차(차량)"가 다를 수 있음을 고려해 driver_id와 car_num을 모두 외래키로 기록하여 데이터의 추적성을 확보했습니다. 도착 시간(arrival_time)은 배차 직후에는 운송이 완료되지 않았으므로 NULL을 허용하고, 운송 완료 시점에 업데이트하도록 설계했습니다.

02

홈화면 및 로그인/회원가입



회원가입 화면입니다. 이 프로젝트의 핵심은 고객과 기사를 구분하는 것이라고 생각했습니다. 그래서 콤보박스를 이용해서 가입 유형을 선택하게 했습니다. 특히 기사님으로 가입할 때는 차량 정보가 필수적으로 들어가야 하는데, 이때 USERS 테이블뿐만 아니라 VEHICLE 테이블에도 동시에 데이터가 들어가야 해서 트랜잭션 처리에 대해 많이 공부하게 되었습니다.

02

홈화면 및 로그인/회원가입



Pick & Go

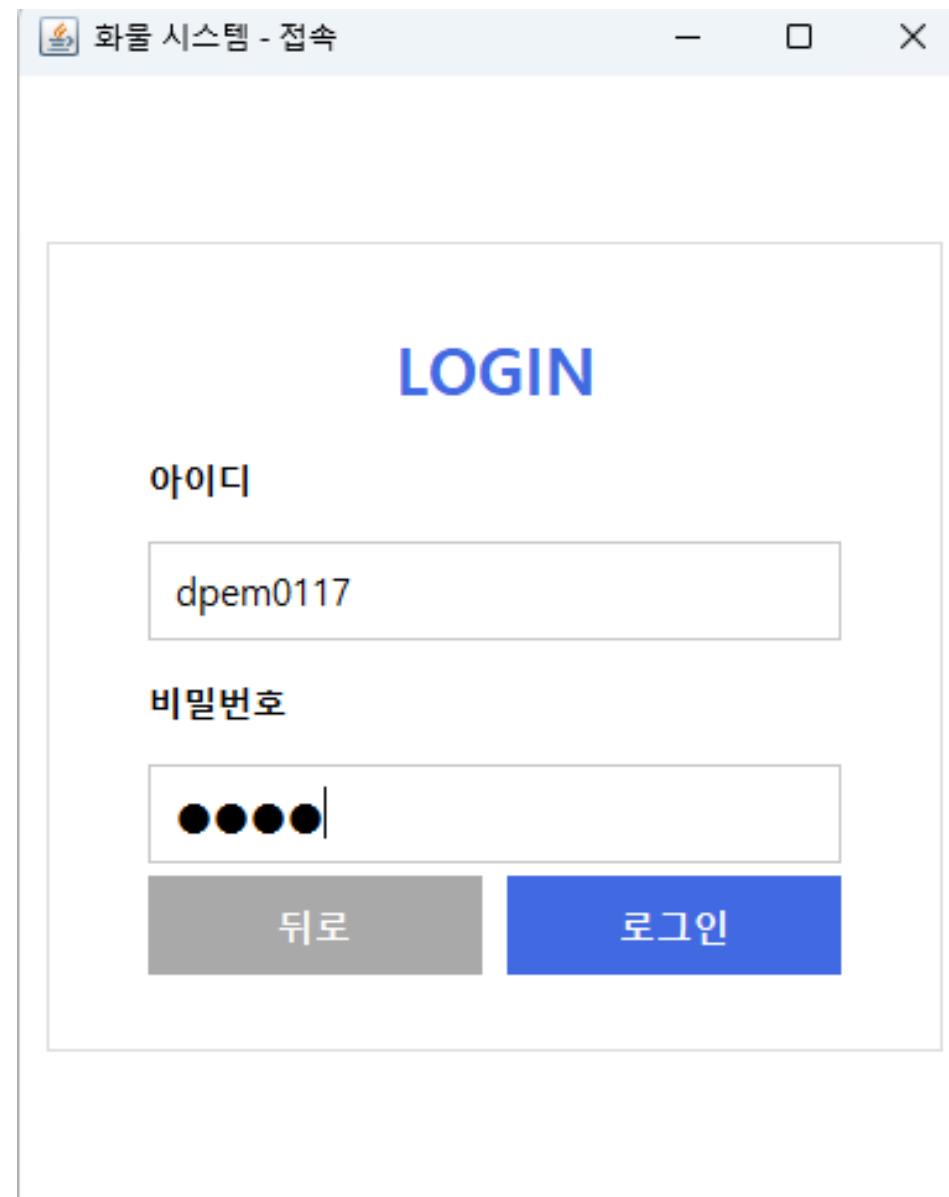
로그인 하러가기

회원가입 하러가기

프로젝트를 실행했을 때 가장 먼저 뜨는 메인 화면입니다. 어떤 기능을 구현할까 고민하다가 화물 배차 시스템인 Pick & Go라는 이름을 지어보았습니다. 처음에는 복잡한 메뉴를 생각했는데, 교수님께서 UI는 직관적인 게 좋다고 하셨던 게 기억나서 로그인과 회원가입 버튼만 심플하게 배치했습니다. Swing의 기본 디자인이 너무 옛날 느낌이라서 버튼 색상이나 폰트를 깔끔하게 보이도록 신경 썼습니다.

02

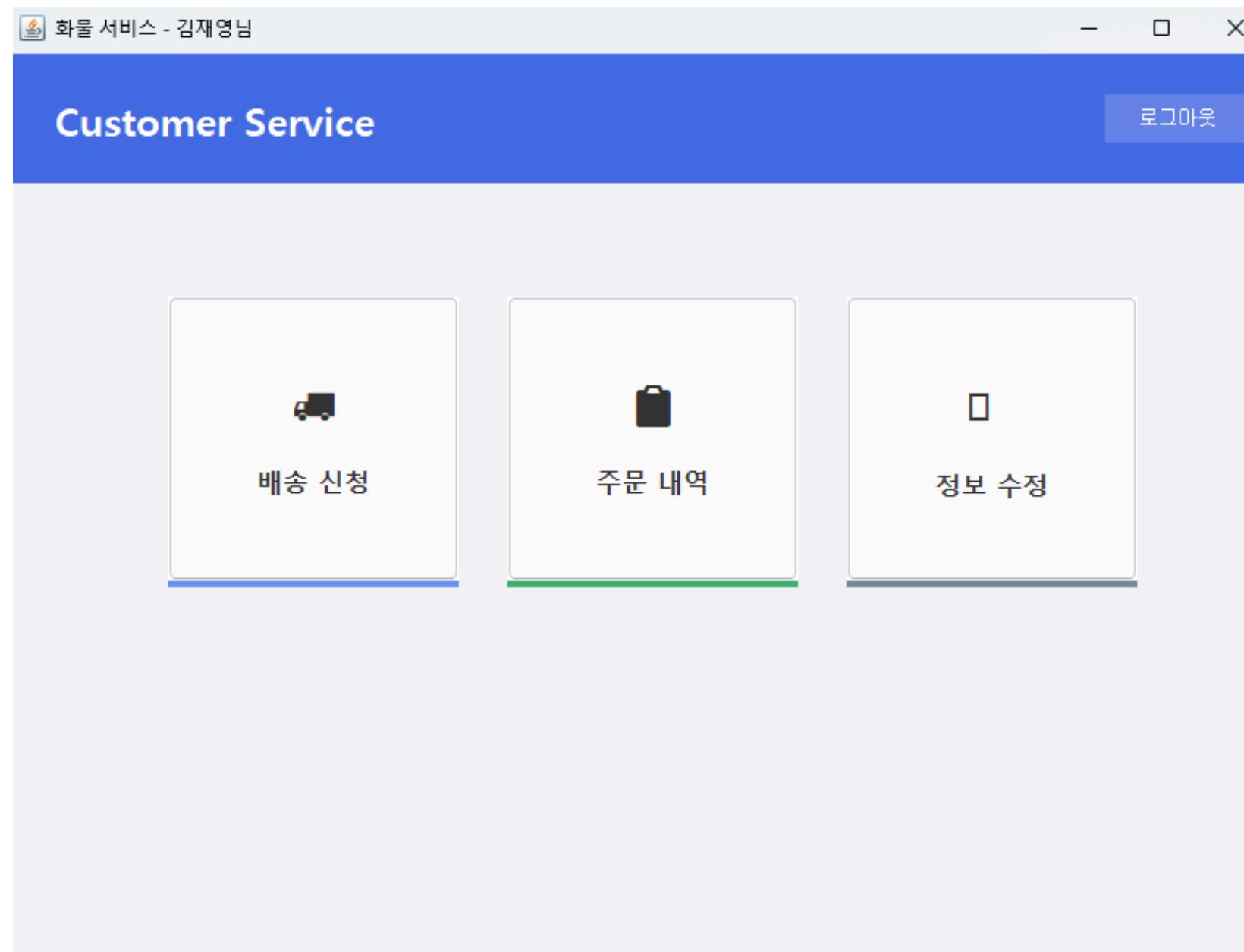
홈화면 및 로그인/회원가입



고객 로그인 화면입니다. 사용자가 아이디와 비밀번호를 입력하면 DB에 있는 정보와 비교합니다. 수업 시간에 배운 PreparedStatement를 사용해서 보안성을 높였고, 로그인에 성공하면 사용자 타입(고객/기사)을 확인해서 각각 다른 대시보드로 이동하도록 로직을 짰습니다. 엔터 키를 치면 바로 로그인이 되도록 편의성도 추가해 보았습니다.

03

고객 홈화면



로그인 후 들어오는 고객용 메인 대시보드입니다. 화면이 계속 바뀌는 것을 어떻게 구현할지 막막했는데, CardLayout이라는 것을 알게 되어 적용했습니다. 배송 신청, 주문 내역, 정보 수정 기능을 한 화면에서 전환되도록 만들었고, 아이콘을 넣어서 좀 더 실제 프로그램처럼 보이게 디자인했습니다.

03

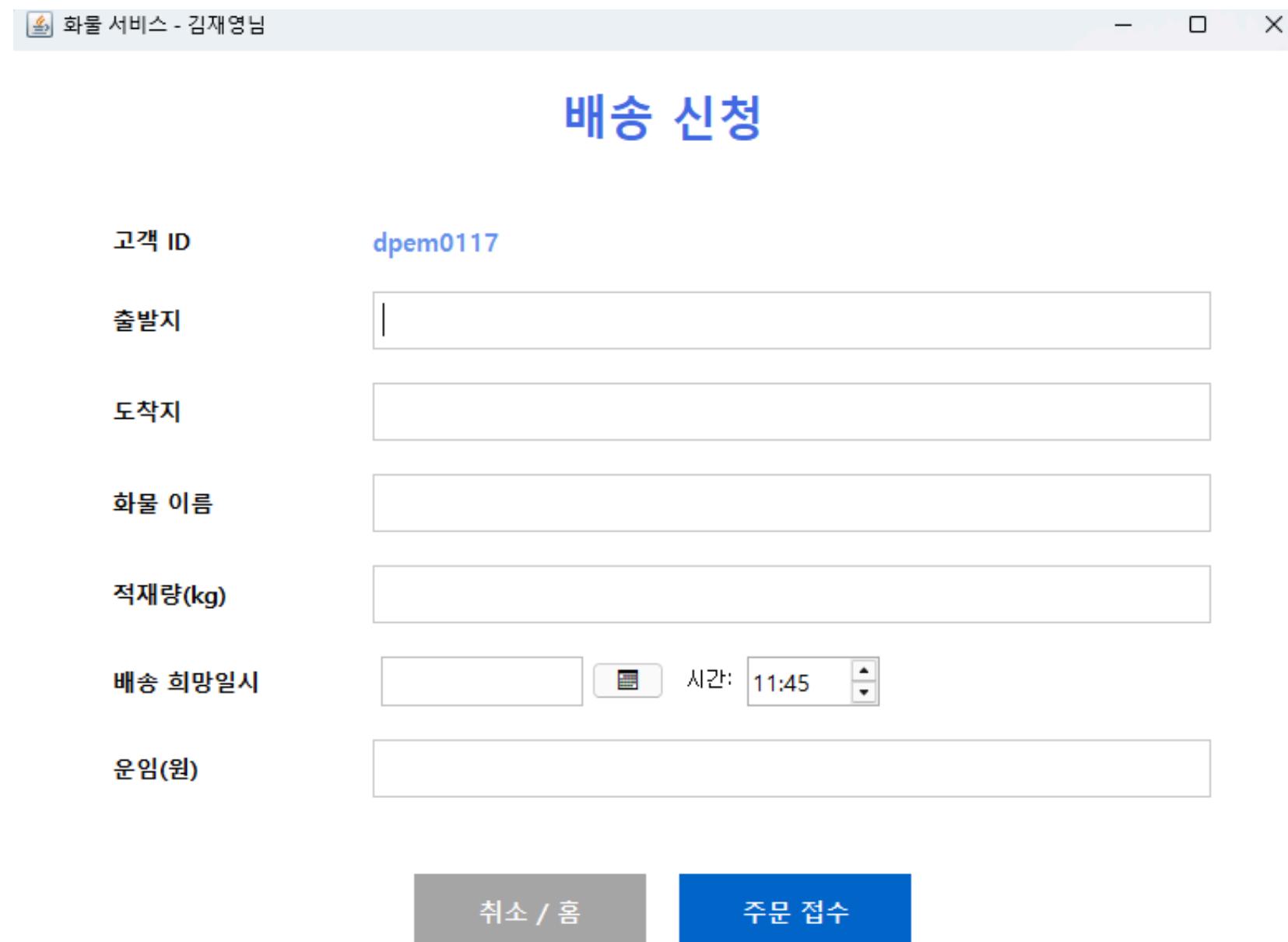
고객홈화면



이건 제가 UI를 만들면서 가장 공들인 부분 중 하나인 로그아웃 버튼입니다. 처음에는 마우스를 올리면 버튼이 하얗게 변해서 글씨가 안 보이는 문제가 있었습니다. 구글링을 해보니 Swing 버튼의 기본 속성이 때문이었습니다. 그래서 paintComponent 메서드를 오버라이딩해서 마우스를 올려도 배경색이 투명하게 유지되도록 코드를 수정했습니다. 기능뿐만 아니라 사용자가 보기 편한 디자인도 중요하다는 것을 느꼈습니다.

04

고객-배송신청(Insert)



화물 서비스 - 김재영님

배송 신청

고객 ID: dpem0117

출발지:

도착지:

화물 이름:

적재량(kg):

배송 희망일시: 시간: 11:45

운임(원):

화물 배송을 신청하는 입력 폼입니다. 고객이 출발지와 도착지, 화물 정보를 입력하는 곳입니다. DB 테이블 설계할 때 정해둔 컬럼들에 맞춰서 텍스트 필드를 배치했습니다.

04

고객-배송신청(Insert)

화물 서비스 - 김재영님

배송 신청

고객 ID	djem0117
출발지	서울정수플리텍
도착지	<input type="text"/>
화물 이름	<input type="text"/>
적재량(kg)	<input type="text"/>
배송 희망일시	<input type="text"/> 시간: 11:45
운임(원)	<input type="text"/>

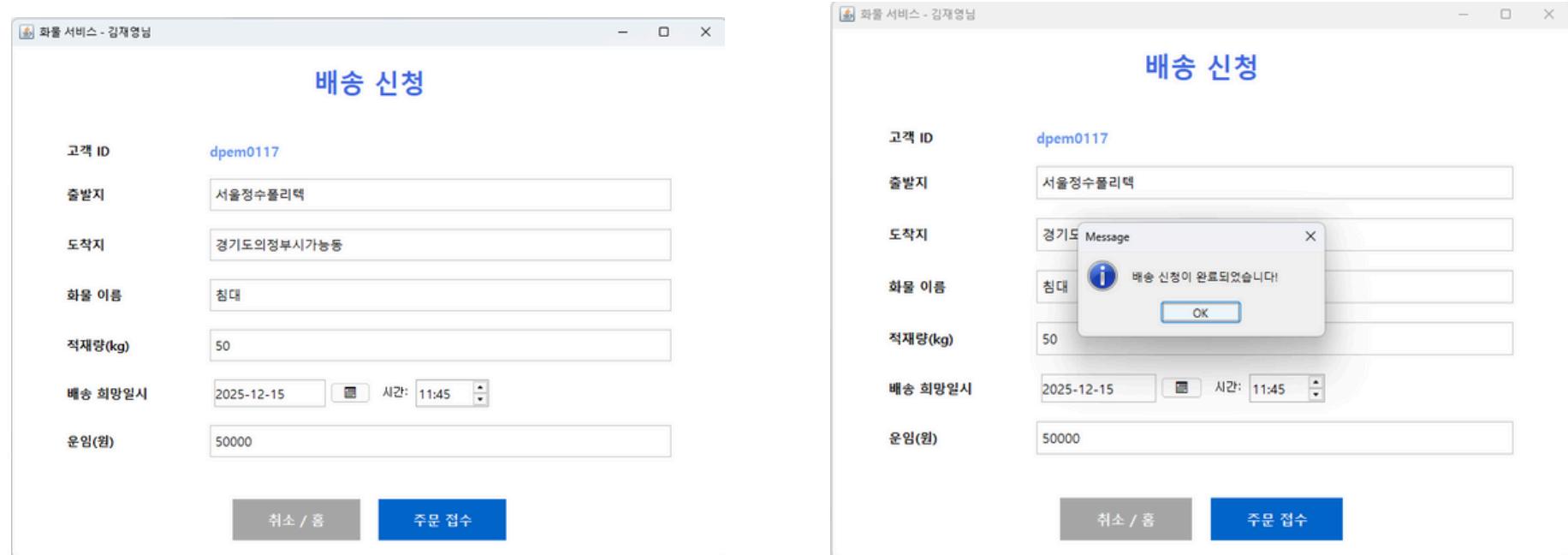
날짜 선택
일 월 화 수 목 금 토
1 2 3 4 5 6
7 8 9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 31
<< 미전 달 2025년 12월 다음 달 >>

취소 / 홈 주문 접수

배송 일시를 입력받을 때 사용자가 직접 날짜를 타이핑하게 하면 오타가 많이 날 것 같았습니다. 그래서 DatePicker라는 기능을 찾아서 적용했습니다. 달력 아이콘을 누르면 팝업창이 뜨고 날짜를 클릭하면 자동으로 입력됩니다. 시간 선택도 JSpinner를 사용해서 분 단위로 정확하게 입력받을 수 있게 구현했습니다.

04

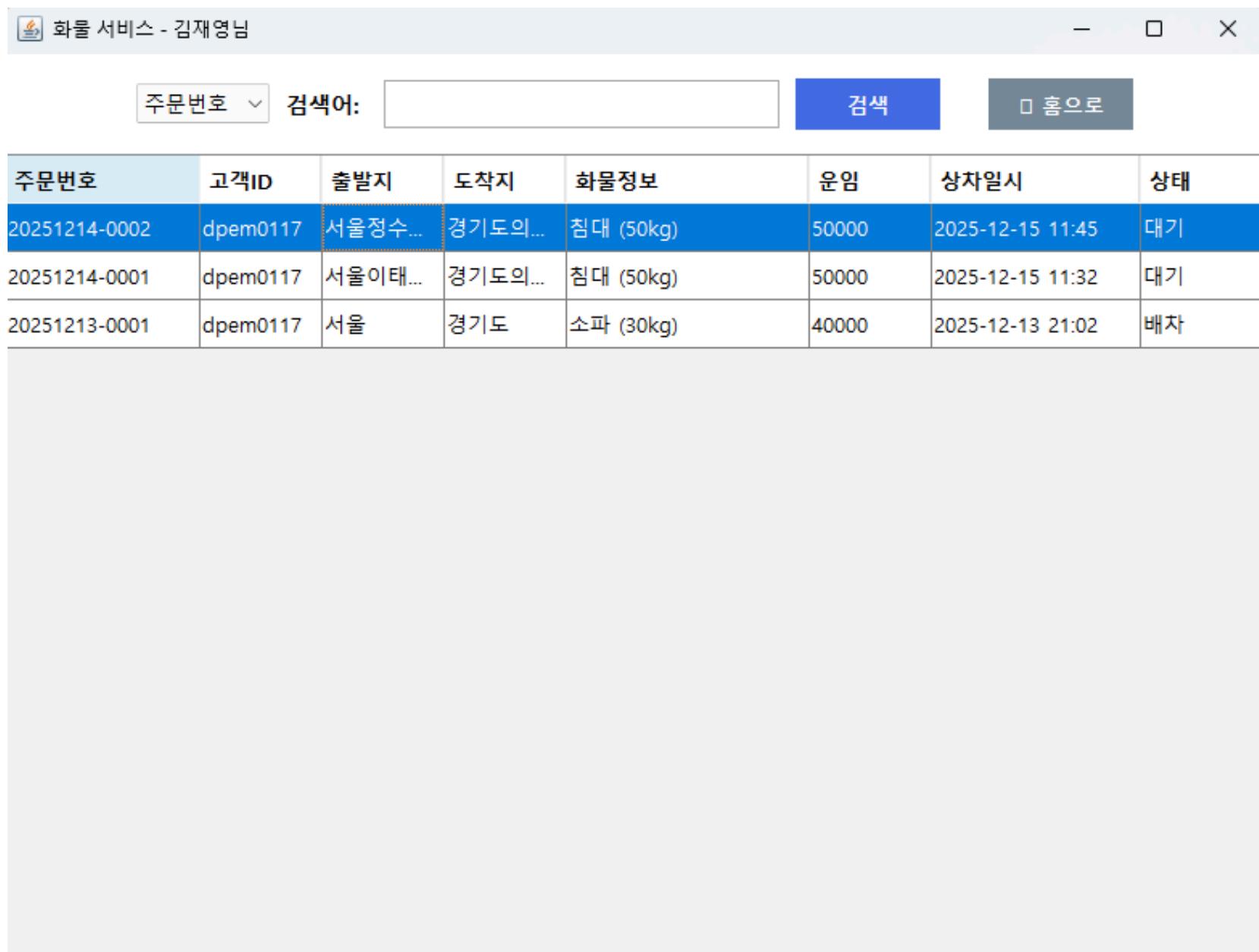
고객-배송신청(Insert)



데이터를 모두 입력하고 주문접수 되었습니다. 여기서 주문 접수 버튼을 누르면 입력된 값들이 VO 객체에 담겨서 DB로 이동합니다. 날짜는 문자열로 받고 DB에는 Date 타입으로 저장해야 해서, 중간에 포맷을 변환하는 과정이 필요했습니다. 이 부분에서 데이터 타입 불일치 에러가 종종 났었는데, 형 변환을 꼼꼼하게 처리해서 해결했습니다.

04

고객-배송신청(Insert)



주문 등록 후 DB에 있는 데이터를 불러온 화면입니다. 단순히 INSERT만 하고 끝나는 게 아니라, SELECT 쿼리를 날려서 실제 ORDER_SHEET 테이블에 들어간 데이터를 JTable로 시각화했습니다. 오라클 DB에 저장된 데이터가 자바 GUI 화면에 그대로 뜨는 걸 보니 연동이 잘되었다는 확신이 들었습니다.

05

고객-주문내역(Search)

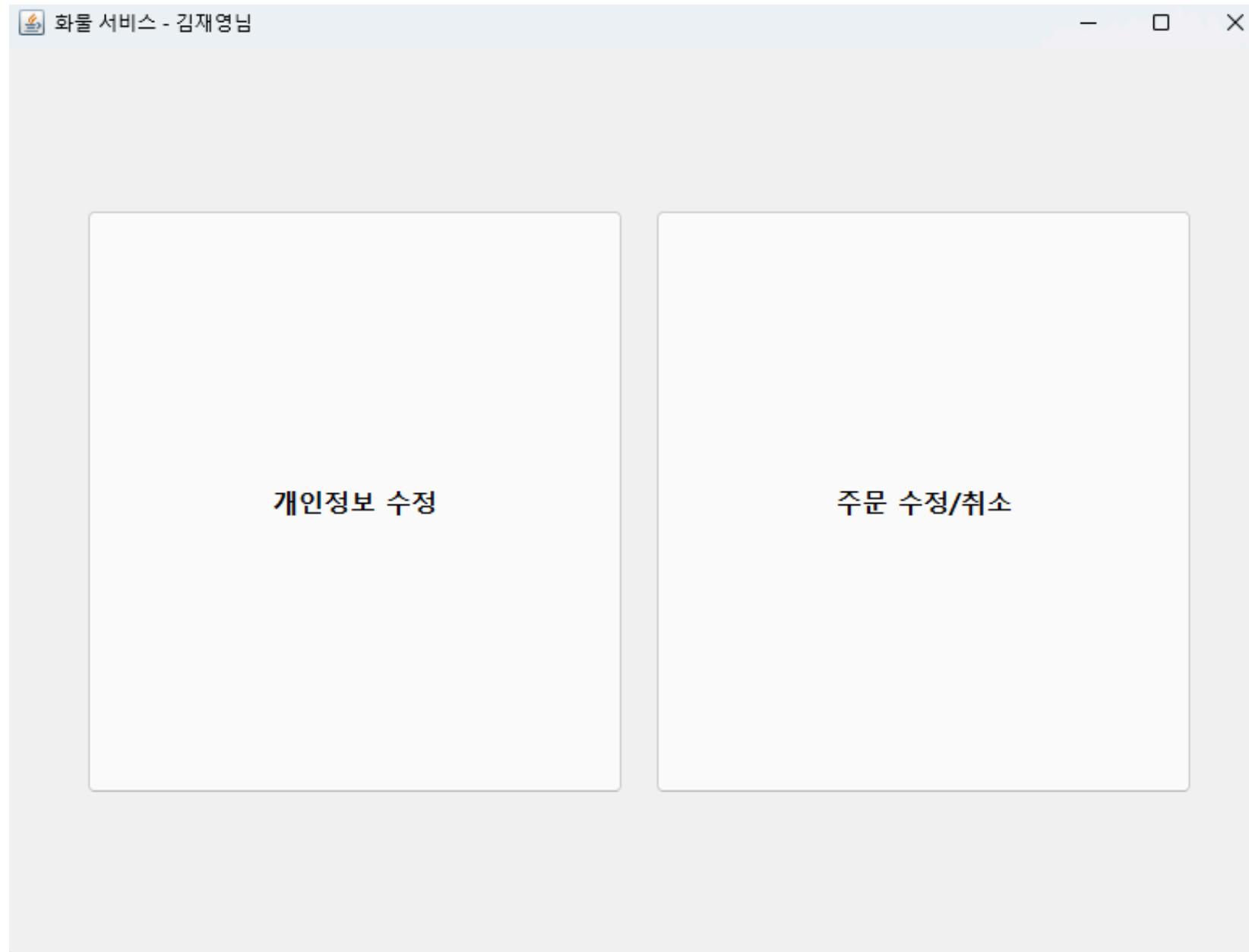


이건 주문 내역 조회 화면의 전체적인 모습입니다. 상단에 검색 기능을 넣어서 주문번호나 출발지로 필터링할 수 있게 만들었습니다. 특히 상차일시 부분은 DB에서 가져올 때 TO_CHAR 함수를 써서 보기 좋게 포맷팅했습니다. 데이터가 쌓여도 스크롤을 통해 확인할 수 있도록 JScrollPane을 적용했습니다.

06

고객-내정보수정 및 주문수정/취소

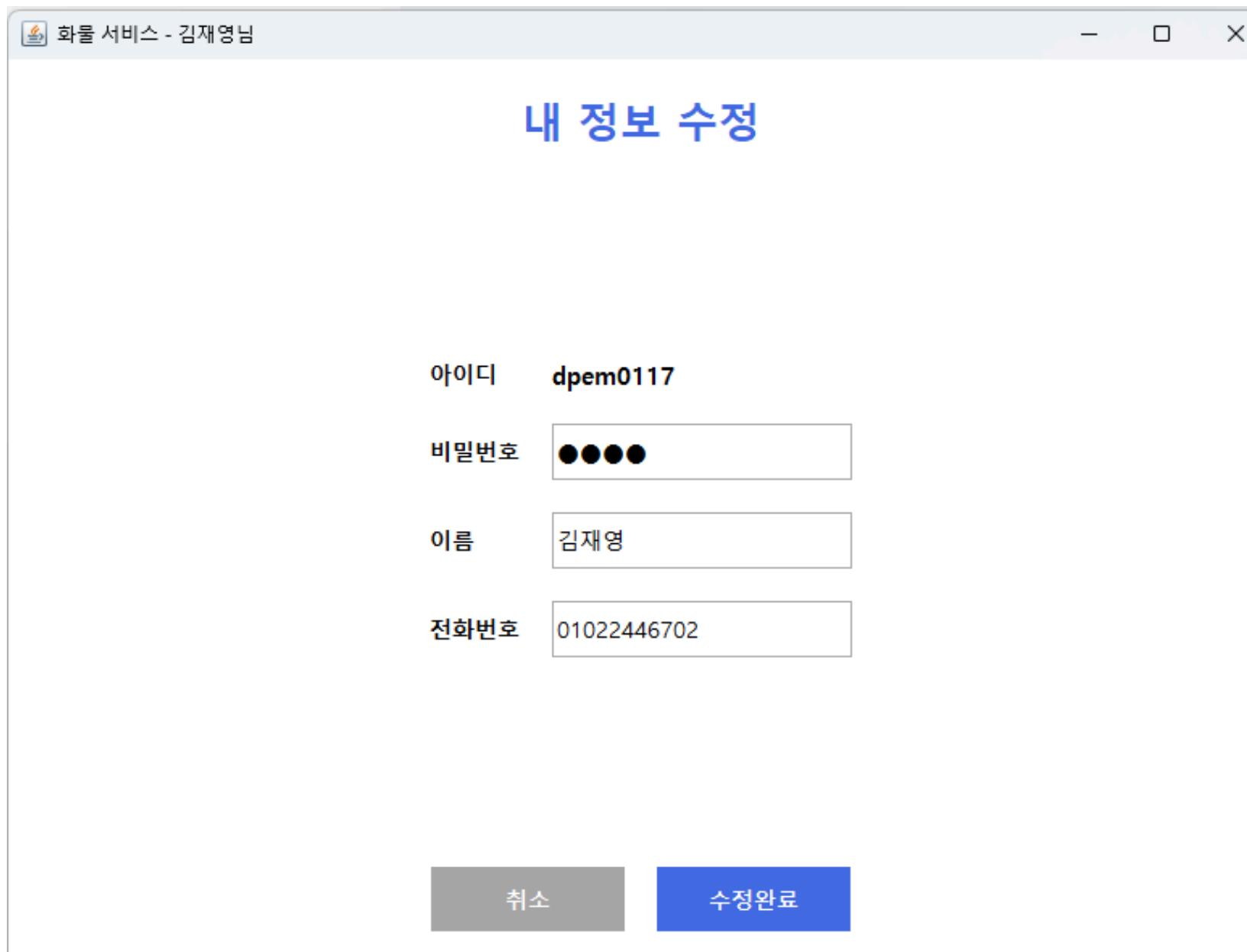
(Update/Delete)



정보 수정 메뉴입니다. 처음에는 버튼 하나로 다 처리하려고 했는데, 개인정보(비밀번호, 전화번호)를 바꾸는 것과 화물 주문 내역을 수정하는 건 성격이 다르다고 생각했습니다. 그래서 사용자가 헷갈리지 않게 두 개의 큰 버튼으로 나누어 메뉴를 구성했습니다. CardLayout을 활용해서 화면 전환이 매끄럽게 되도록 구현했습니다.

06

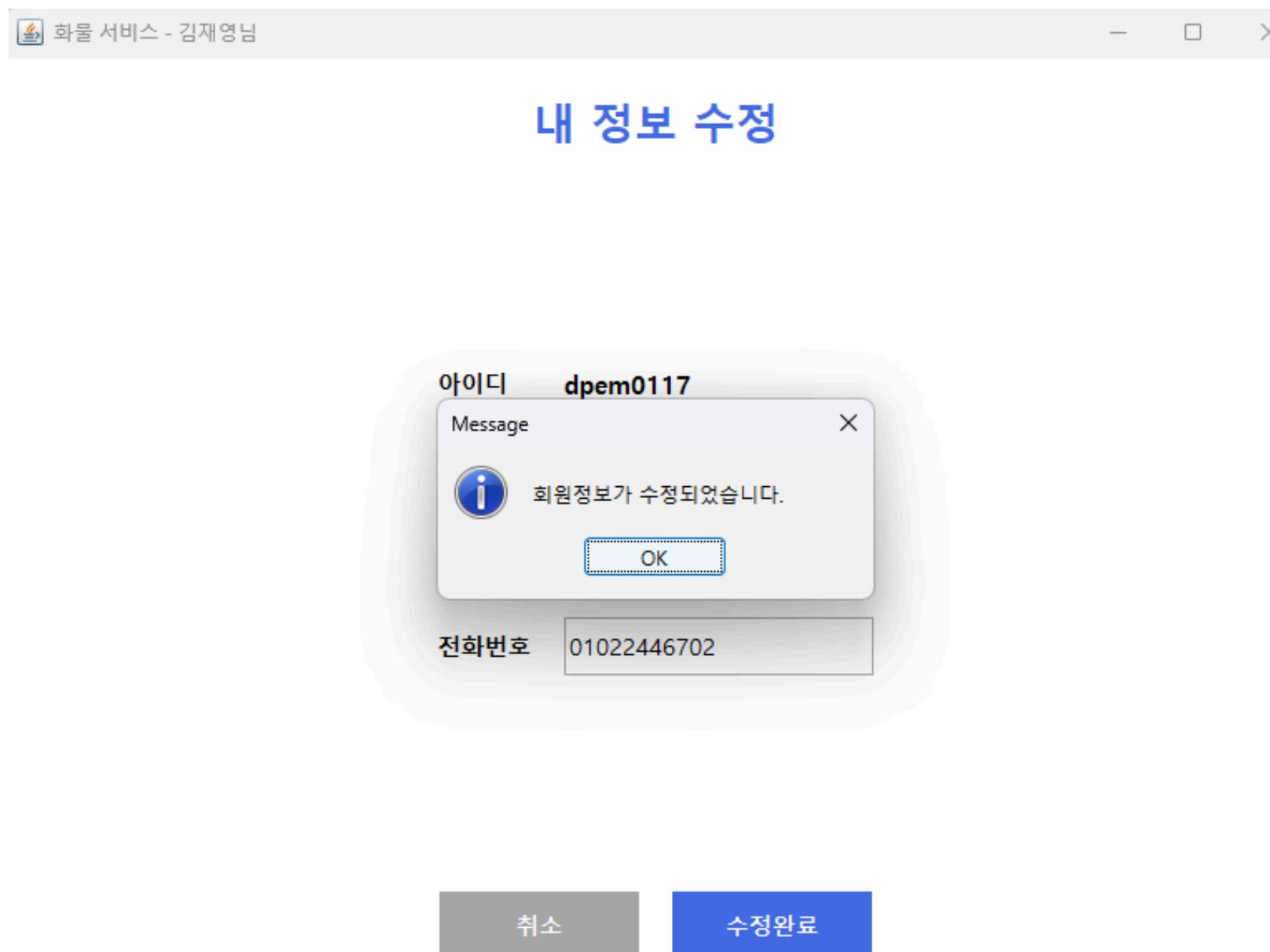
고객-내정보수정 및 주문수정/취소 (Update/Delete)



회원 개인정보를 수정하는 화면입니다. 로그인한 사용자의 정보를 가져와서 텍스트 필드에 미리 뿌려주도록 만들었습니다. 아이디는 PK(기본키)라서 수정하면 안 되기 때문에 JLabel로 처리하거나 수정 불가로 막았고, 비밀번호나 전화번호 같은 변경 가능한 정보만 입력받도록 설계했습니다.

06

고객-내정보수정 및 주문수정/취소 (Update/Delete)

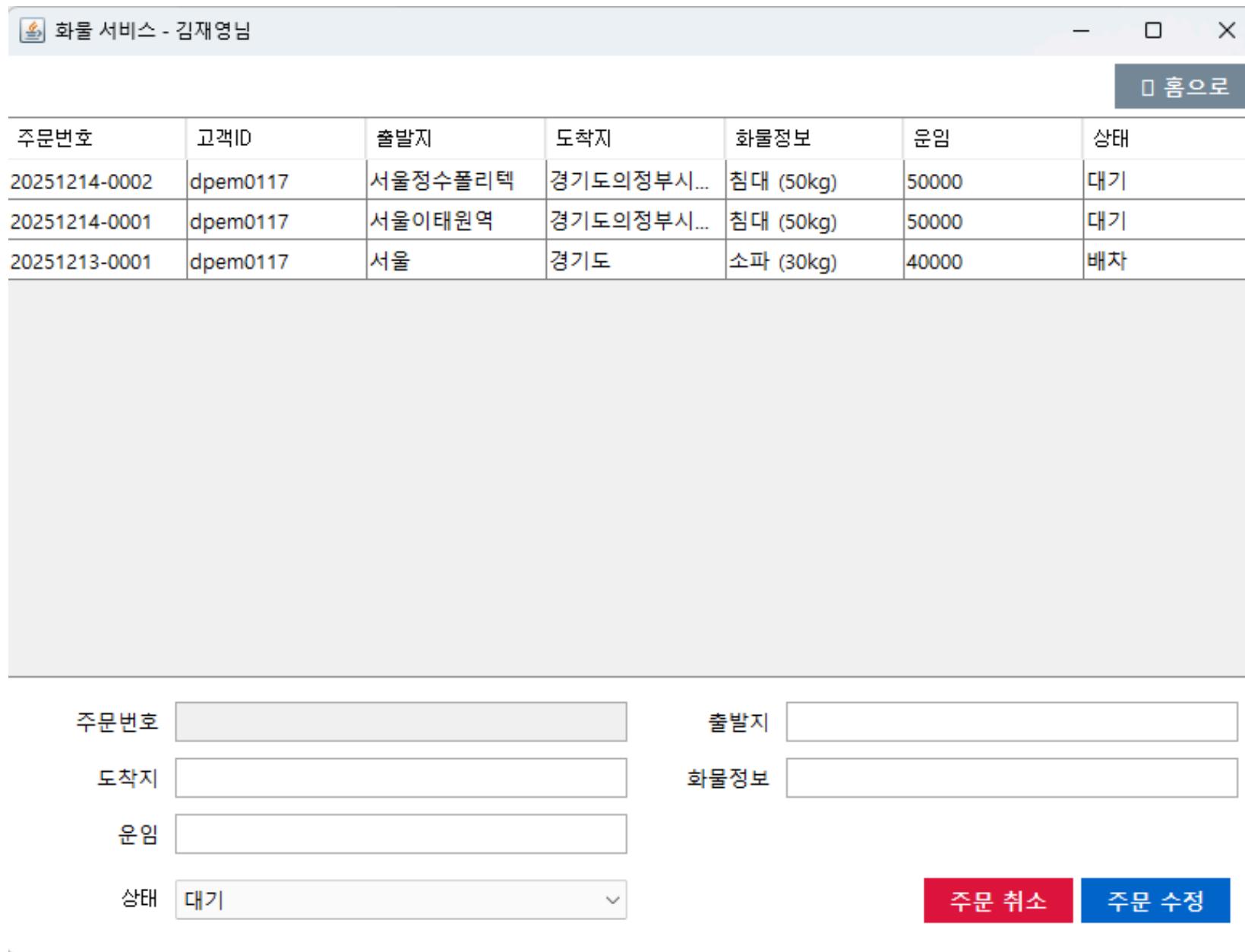


수정 완료 버튼을 눌렀을 때의 모습입니다. 이때 내부적으로는 UPDATE USERS SET... 쿼리가 실행됩니다. 쿼리가 성공적으로 수행되면 1을 반환하는데, 이걸 체크해서 사용자에게 수정되었다는 피드백을 주는 팝업창을 띠웠습니다. 이런 사소한 알림이 있어야 사용자가 안심할 수 있다고 생각했습니다.

06

고객-내정보수정 및 주문수정/취소

(Update/Delete)



주문 수정 및 취소를 할 수 있는 화면입니다. 상단에는 테이블이 있고 하단에는 입력 폼이 있습니다. 여기서 제일 신경 쓴 부분은 편의성입니다. 마우스로 테이블의 특정 행을 클릭하면, MouseListener가 작동해서 해당 행의 데이터가 아래쪽 텍스트 필드에 자동으로 채워지도록 만들었습니다. 일일이 주문번호를 칠 필요가 없어서 훨씬 편해졌습니다.

06

고객-내정보수정 및 주문수정/취소 (Update/Delete)

화물 서비스 - 김재영님

홈으로

주문번호	고객ID	출발지	도착지	화물정보	운임	상태
20251214-0002	djem0117	서울정수플리텍	경기도의정부시...	침대 (50kg)	50000	대기
20251214-0001	djem0117	서울이태원역	경기도의정부시...	침대 (50kg)	50000	대기
20251213-0001	djem0117	서울	경기도	소파 (30kg)	40000	배차

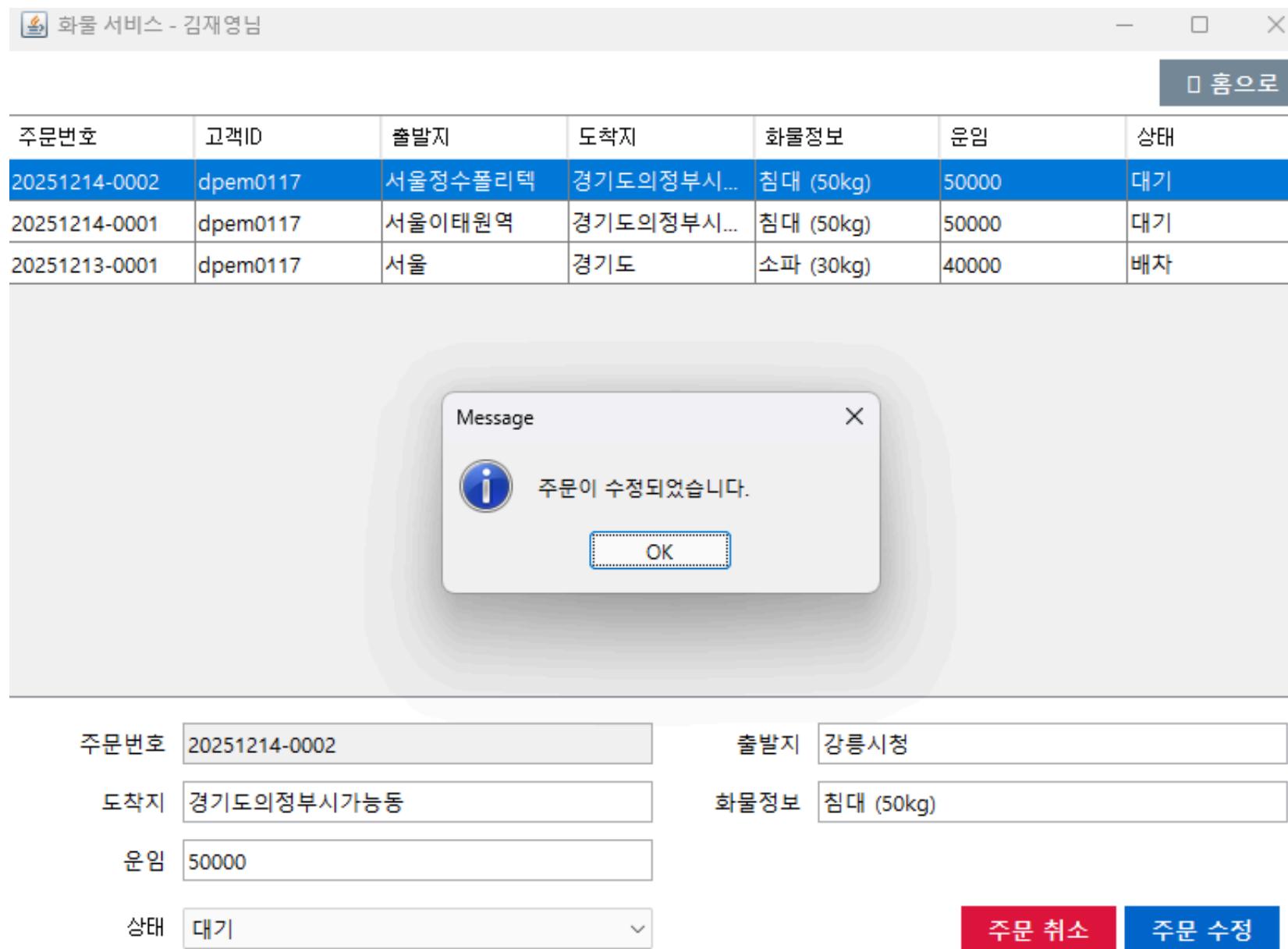
주문번호: 20251214-0002 출발지: 강릉시청
도착지: 경기도의정부시가능동 화물정보: 침대 (50kg)
운임: 50000 상태: 대기

주문 취소 주문 수정

데이터 수정(Update) 기능을 테스트하는 모습입니다.
기존에 서울정수플리텍으로 되어있던 출발지를 강릉시청
으로 바꾸고, 운임이나 화물 정보도 변경할 수 있습니다.
텍스트 필드에서 내용을 고치고 주문 수정 버튼을 누르면
DB의 내용이 갱신됩니다.

06

고객-내정보수정 및 주문수정/취소 (Update/Delete)



수정 버튼을 누르고 DB 업데이트가 성공했을 때 뜨는 메시지입니다. JDBC 프로그래밍을 하면서 예외 처리가 정말 중요하다는 걸 느꼈는데, 만약 DB 오류가 났다면 여기에 에러 메시지가 떴을 겁니다. 정상적으로 완료되었다는 창이 뜨면, 프로그램이 데이터를 다시 로드해서 화면을 새로고침합니다.

06

고객-내정보수정 및 주문수정/취소

(Update/Delete)

The screenshot shows a Windows application window titled "화물 서비스 - 김재영님". At the top right, there is a button labeled "홈으로". The main area contains a table with columns: 주문번호, 고객ID, 출발지, 도착지, 화물정보, 운임, and 상태. The table has three rows of data:

주문번호	고객ID	출발지	도착지	화물정보	운임	상태
20251214-0002	dpem0117	강릉시청	경기도의정부시...	침대 (50kg)	50000	대기
20251214-0001	dpem0117	서울이태원역	경기도의정부시...	침대 (50kg)	50000	대기
20251213-0001	dpem0117	서울	경기도	소파 (30kg)	40000	배차

Below the table, there is a form with fields for updating an order:

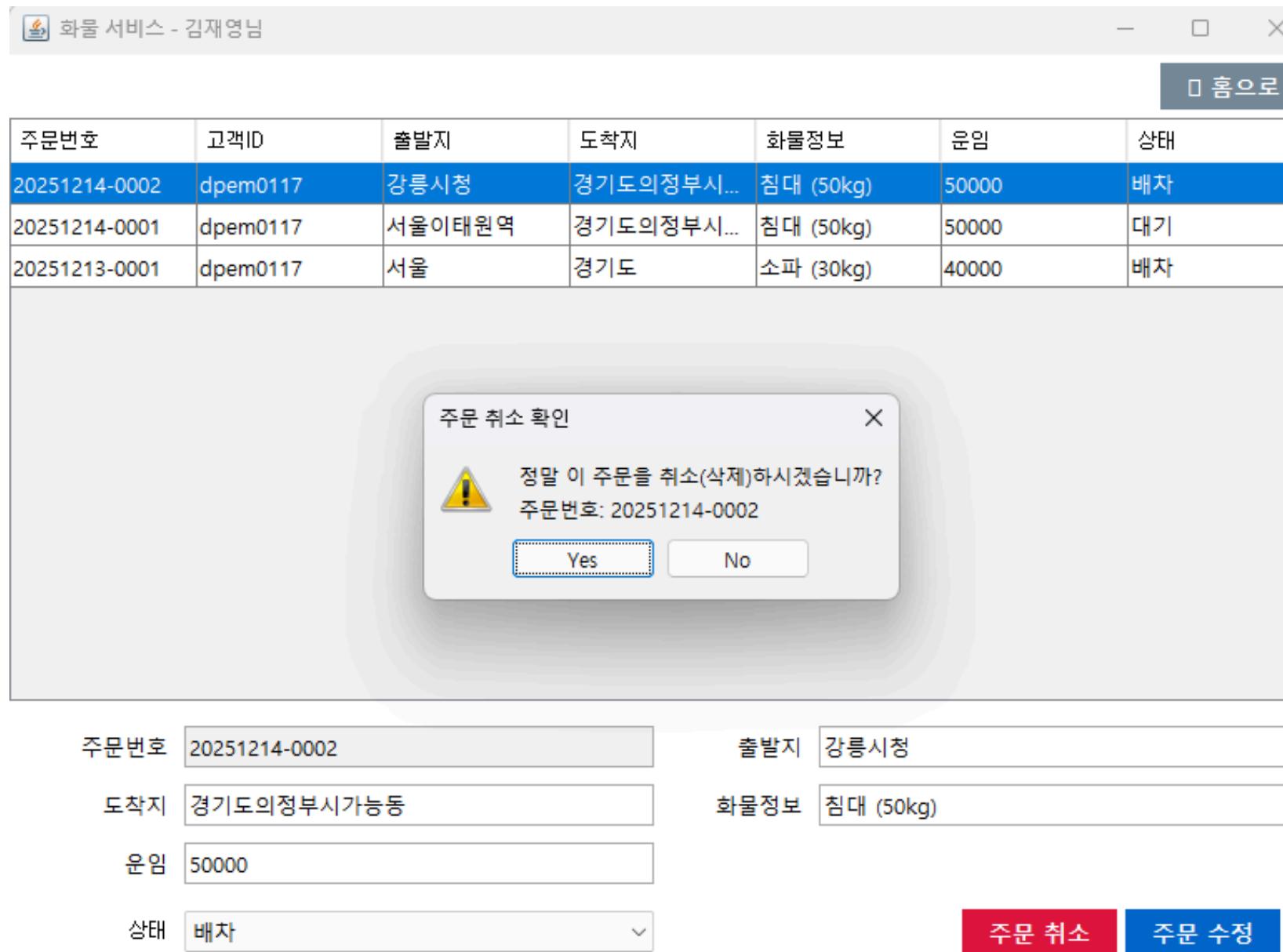
주문번호	20251214-0002	출발지	강릉시청
도착지	경기도의정부시가능동	화물정보	침대 (50kg)
운임	50000		
상태	대기		

At the bottom right of the form, there are two buttons: "주문 취소" (Order Cancel) and "주문 수정" (Order Update).

수정이 완료된 후의 테이블 모습입니다. 아까 서울정수풀 리텍이었던 출발지가 강릉시청으로 즉시 변경된 것을 볼 수 있습니다. 프로그램을 껐다 켜지 않아도 변경 사항이 바로 반영되도록 테이블 모델을 갱신하는 로직을 넣었습니다. CRUD 중 Update와 Read가 완벽하게 연동되는 것을 확인했습니다.

06

고객-내정보수정 및 주문수정/취소 (Update/Delete)

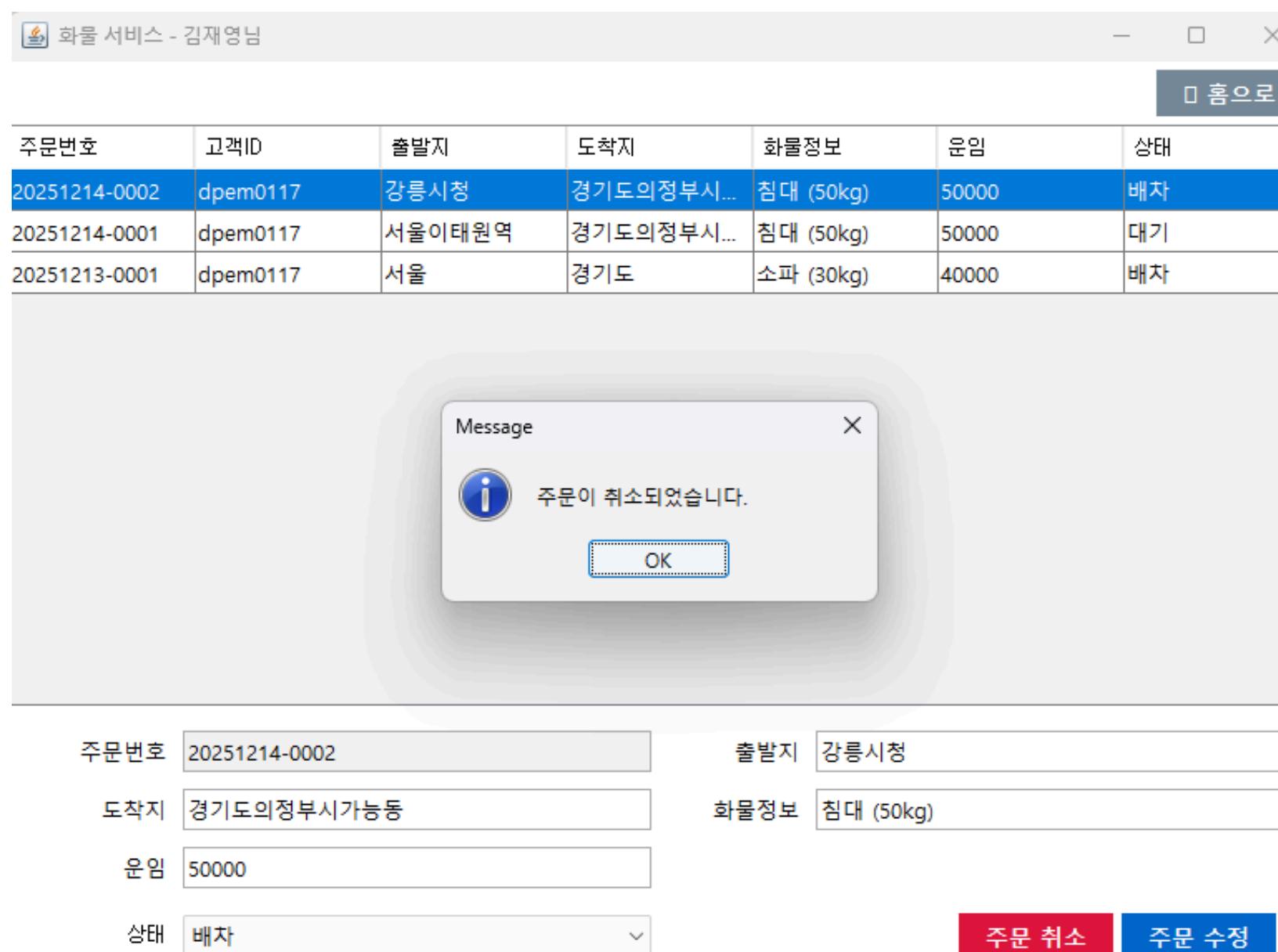


마지막으로 삭제(Delete) 기능입니다. 주문 취소 버튼을 누르면 바로 지워지는 건 너무 위험하다고 생각해서, JOptionPane을 사용해 확인 팝업을 한 번 더 띄웠습니다. 여기서 Yes를 눌러야만 DELETE 쿼리가 날아갑니다. 사용자 입장에서 실수로 데이터를 날리지 않게 방어 코드를 짜는 것 또한 개발자의 역할이라는 것을 배웠습니다.

06

고객-내정보수정 및 주문수정/취소

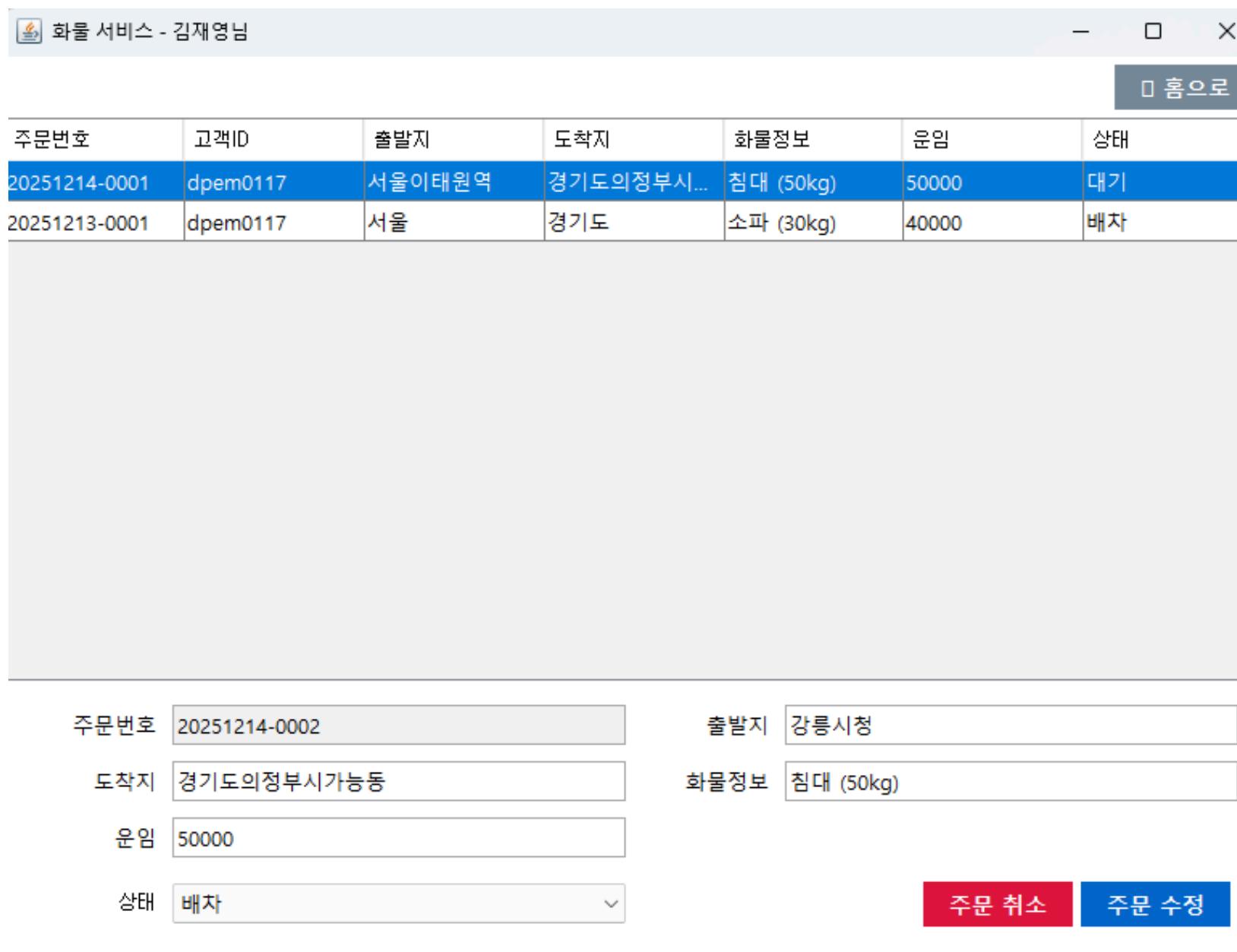
(Update/Delete)



주문이 취소된 직후의 화면입니다. 삭제 쿼리가 성공적으로 수행되면, 자바 코드 내부에서 테이블 모델을 초기화하고 DB에서 데이터를 다시 읽어오는(Select) refresh 메서드가 실행됩니다.

06

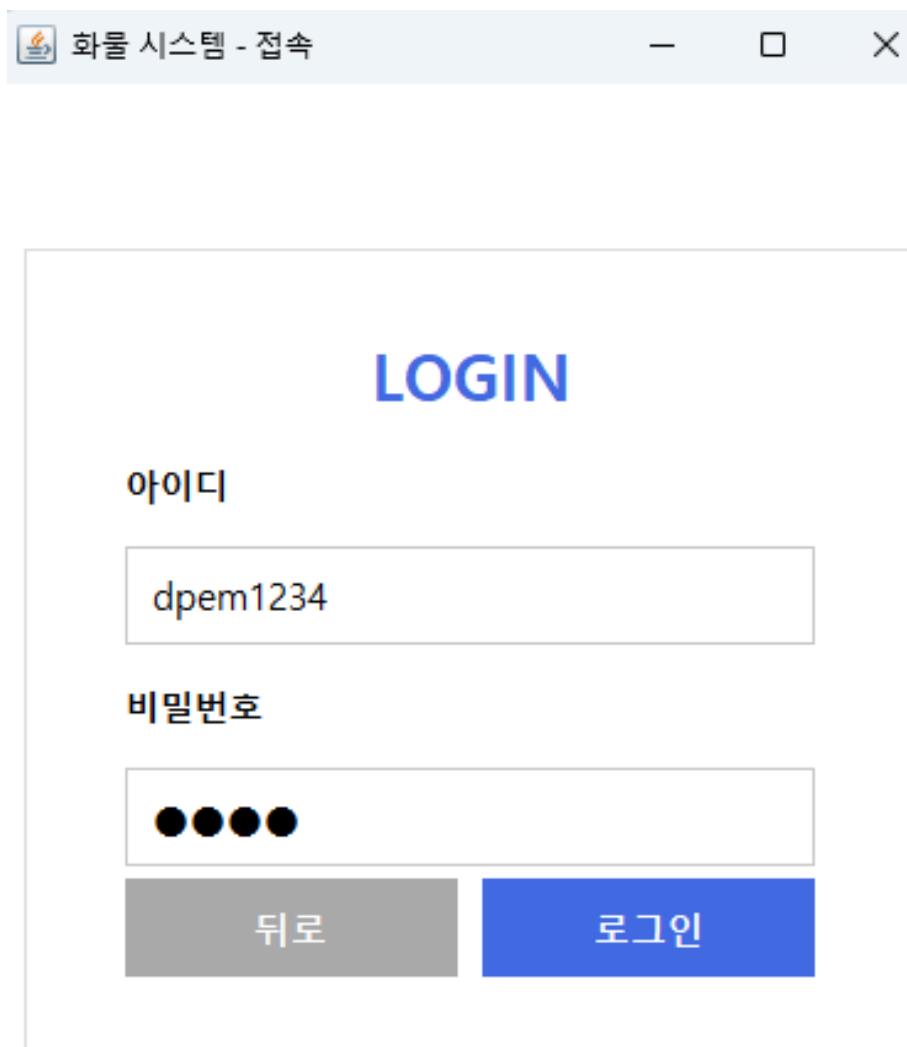
고객-내정보수정 및 주문수정/취소 (Update/Delete)



사용자가 새로고침 버튼을 누르거나 프로그램을 껐다 켜지 않아도, 방금 삭제한 주문이 목록에서 즉시 사라지는 것을 확인할 수 있습니다.

07

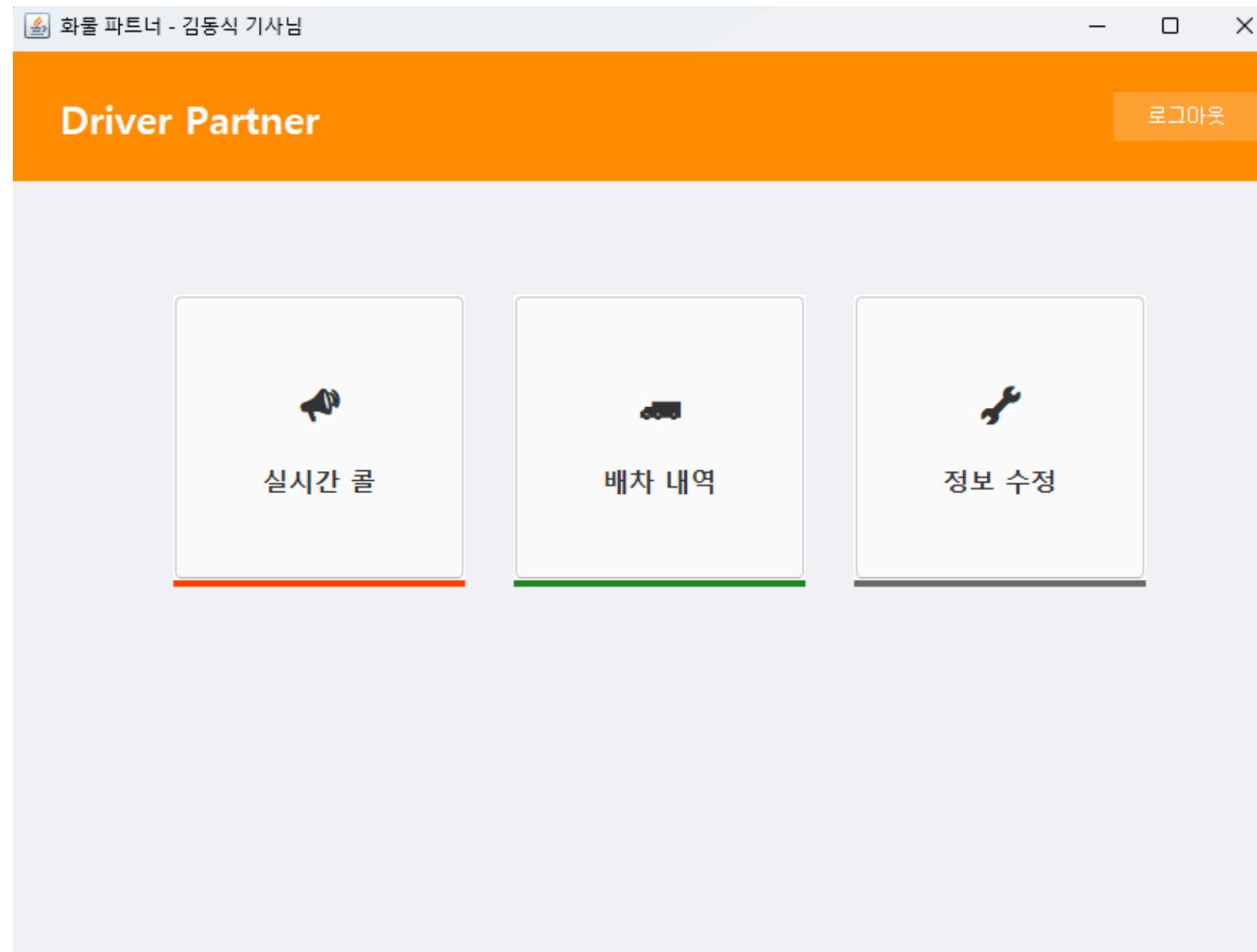
기사 로그인 및 홈화면



이제 기사님(Driver) 기능을 테스트하기 위해 기사 아이디로 로그인하는 모습입니다. 같은 로그인 화면이지만, 내부 로직에서는 사용자가 입력한 아이디의 `user_type` 컬럼을 확인합니다. 타입이 기사(Driver)라면 고객용 파란색 대시보드가 아닌, 기사 전용 주황색 대시보드로 이동하게끔 분기 처리를 구현했습니다.

07

기사 로그인 및 홈화면



기사님 전용 메인 대시보드입니다. 고객용 화면과 차별화를 두기 위해 테마 색상을 주황색(Orange)으로 잡았습니다. 기사님의 업무 흐름에 맞춰 실시간 콜(대기 중인 주문), 배차 관리(내가 잡은 주문), 정보 수정 이렇게 3가지 핵심 기능을 버튼으로 배치했습니다. 여기 있는 로그아웃 버튼도 마찬가지로 마우스 호버 시 배경이 깨지지 않게 커스텀 했습니다.

08

기사-실시간콜 (Search)



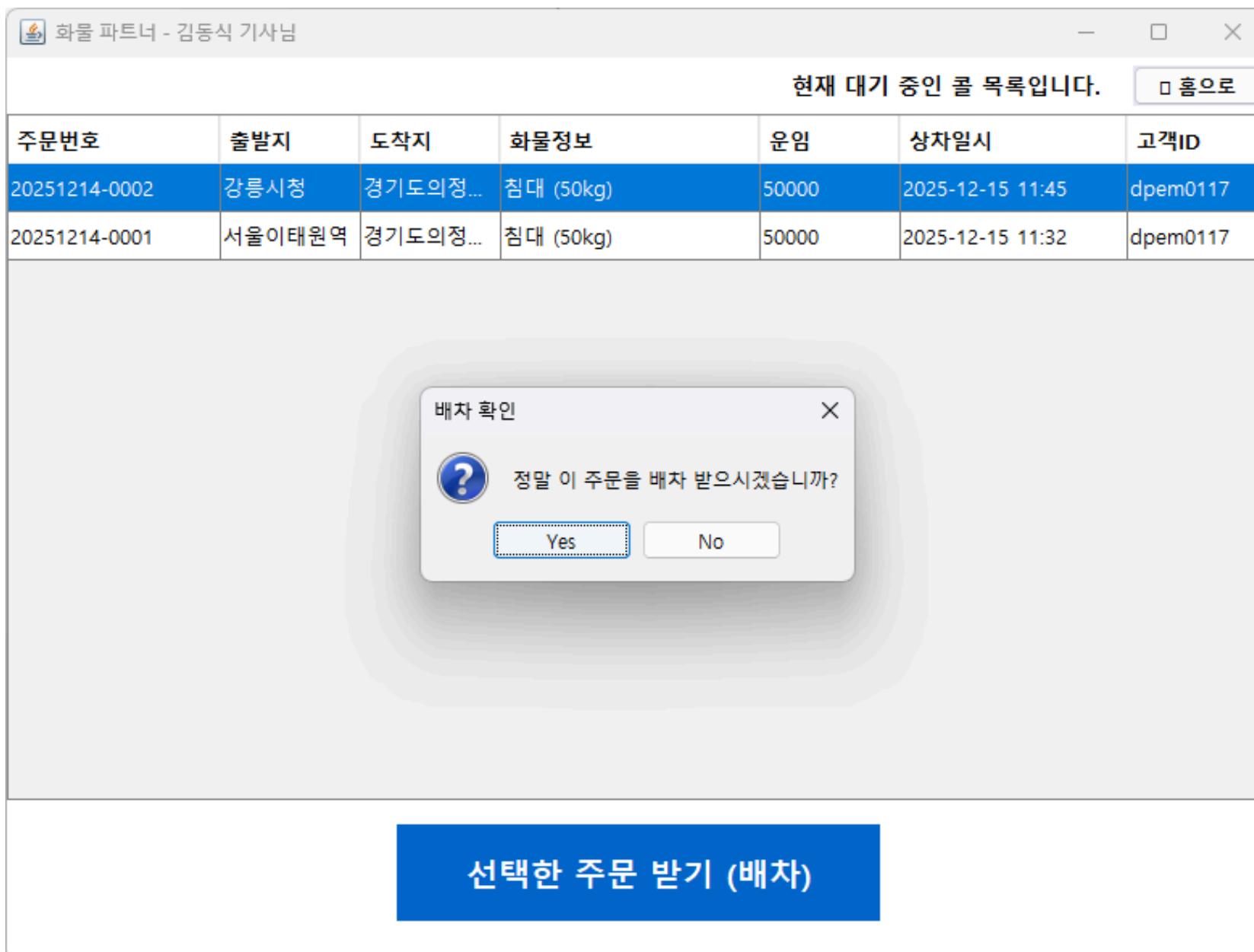
주문번호	출발지	도착지	화물정보	운임	상차일시	고객ID
20251214-0002	강릉시청	경기도의정...	침대 (50kg)	50000	2025-12-15 11:45	dpem0117
20251214-0001	서울이태원역	경기도의정...	침대 (50kg)	50000	2025-12-15 11:32	dpem0117

선택한 주문 받기 (배차)

기사님이 수행하는 가장 중요한 업무인 실시간 콜 조회 화면입니다. 여기서는 ORDER_SHEET 테이블에서 status가 대기인 항목만 필터링해서 SELECT 해옵니다. 이미 다른 기사가 잡은 주문은 뜨지 않아야 하므로 SQL의 WHERE 절 조건을 신경 써서 작성했습니다. 하단의 선택한 주문 받기 버튼을 누르면 배차 로직이 시작됩니다.

08

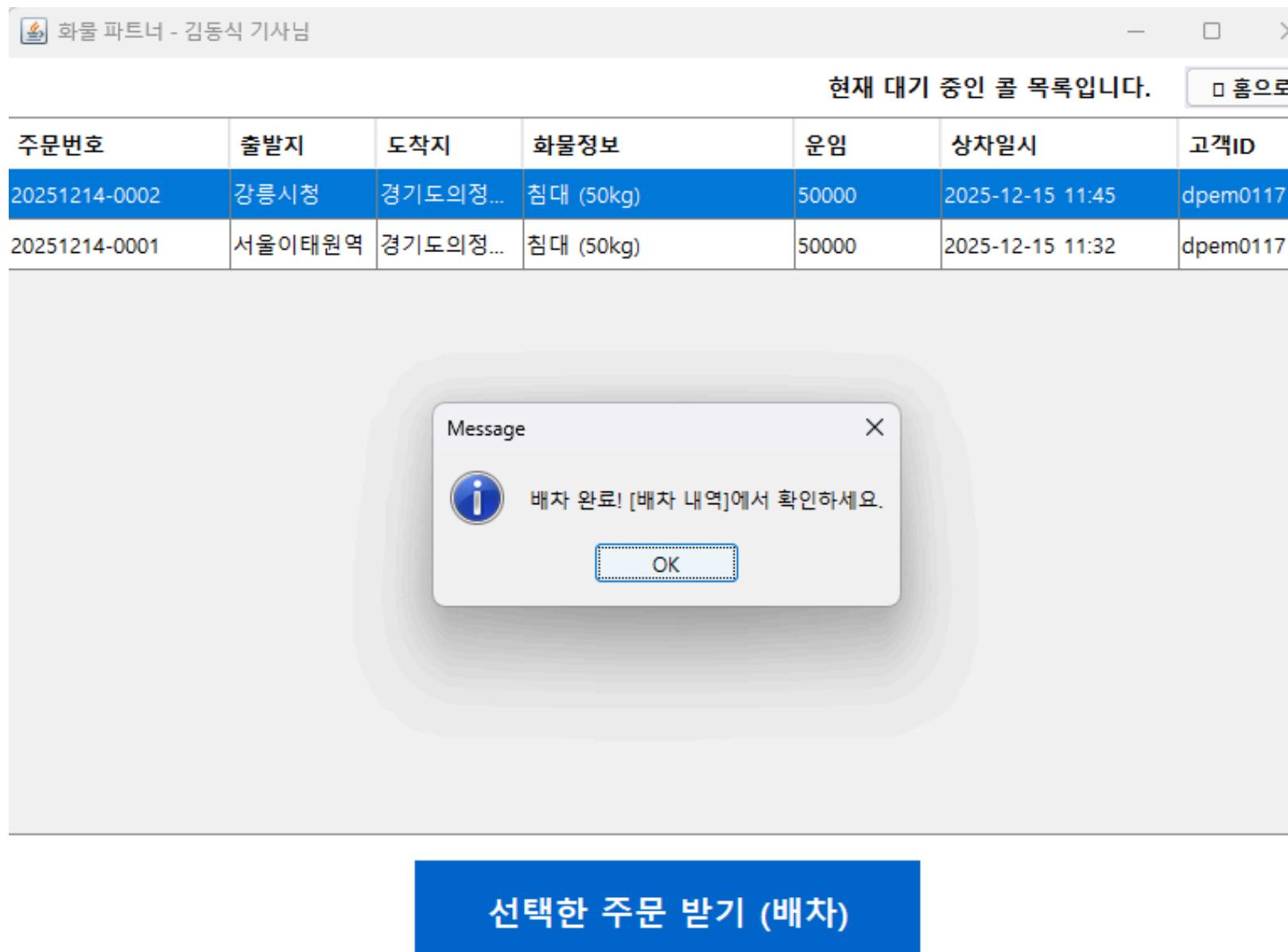
기사-실시간콜 (Search)



주문을 배차받기 전 확인 팝업입니다. 기사님이 실수로 원하지 않는 주문을 클릭했을 수도 있어서 예/아니오를 선택하게 했습니다. 이 과정이 단순해 보이지만, 실제로는 동시성 문제가 발생할 수 있는 부분이라 트랜잭션 처리가 중요하다고 배웠습니다. (이번 과제에서는 JDBC 기본 기능 구현에 집중했습니다.)

08

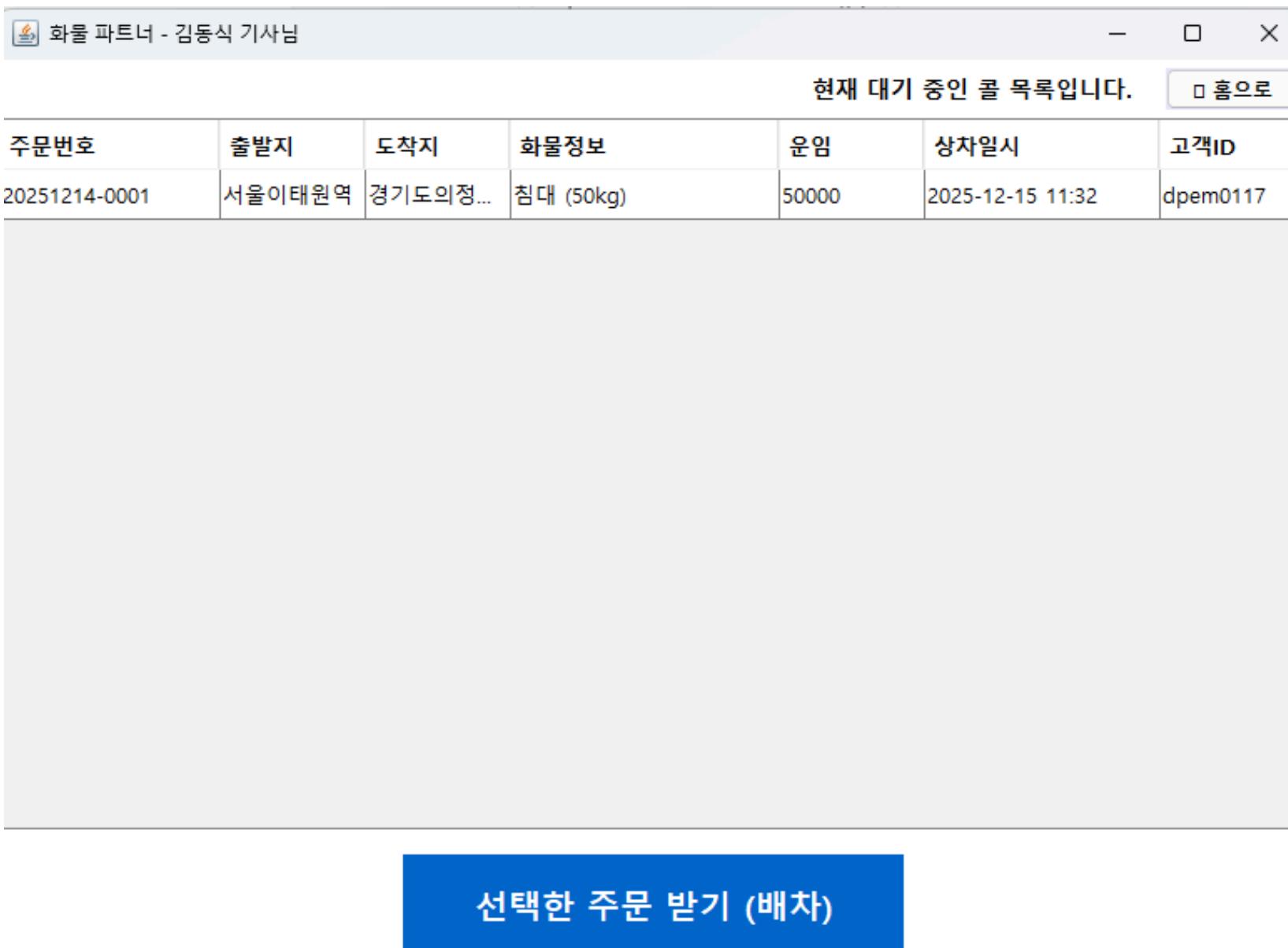
기사-실시간콜 (Search)



배차가 성공적으로 완료되었을 때 뜨는 메시지입니다. 이 확인 버튼을 누르는 순간, 백그라운드에서는 두 가지 DB 작업이 동시에 일어납니다. 첫째는 DISPATCH 테이블에 배차 내역을 INSERT 하는 것이고, 둘째는 ORDER_SHEET 테이블의 상태를 대기에서 배차로 UPDATE 하는 것입니다. 데이터 무결성을 위해 Connection의 setAutoCommit(false)를 설정하여 트랜잭션으로 묶어서 처리했습니다.

08

기사-실시간콜 (Search)



차 완료 후 실시간 콜 목록이 갱신된 모습입니다. 방금 수락한 주문은 상태가 배차로 바뀌었기 때문에, 대기 상태만 보여주는 이 목록에서는 즉시 사라지게 됩니다. 이렇게 화면이 바로바로 반응해야 사용자가 답답해하지 않을 것 같아서 UI 갱신 로직에 신경을 많이 썼습니다.

09

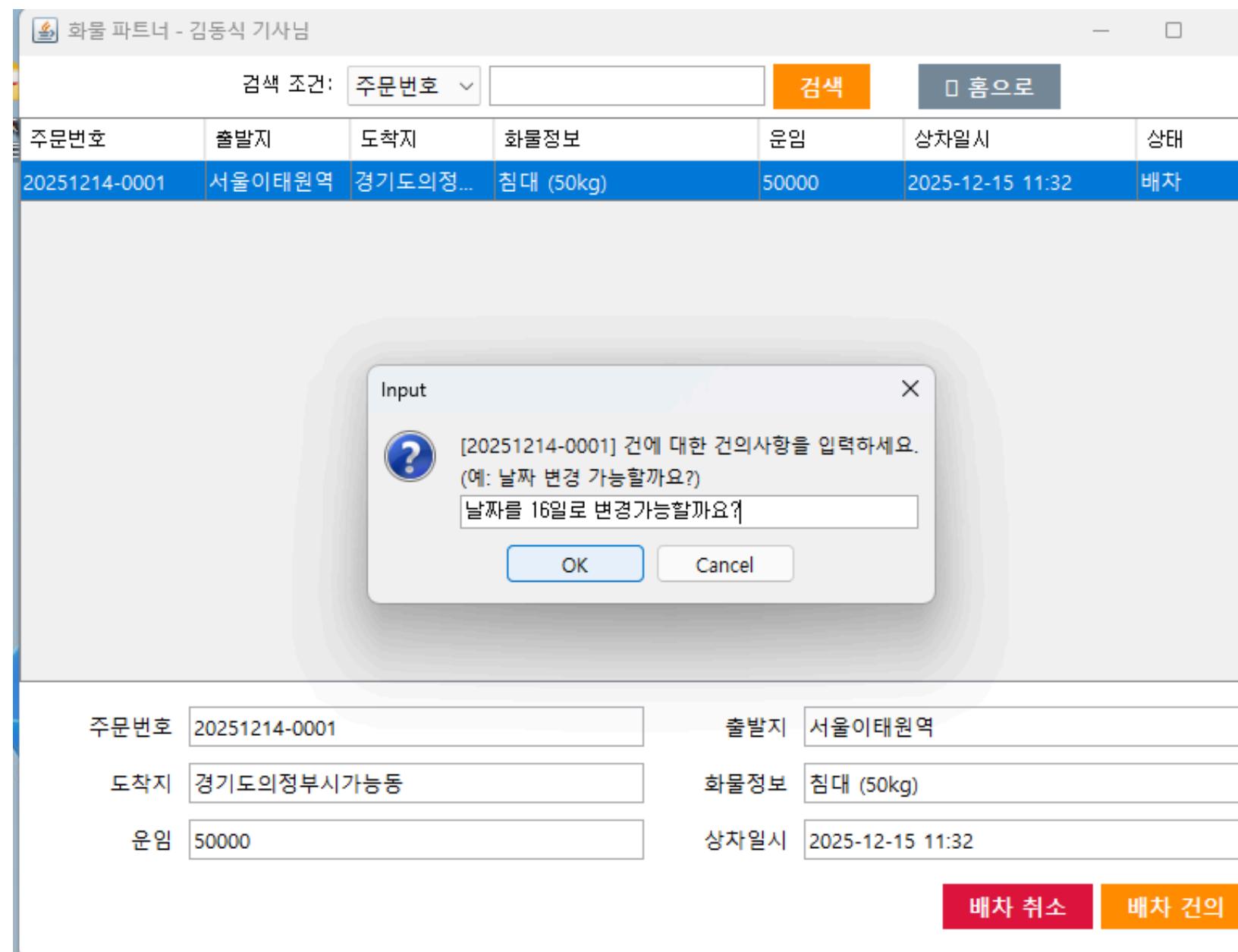
기사-배차내역(Search)/배차관리 (Update/Delete)



기사님의 배차 관리(내역) 화면입니다. 고객의 주문 내역 화면과 레이아웃은 비슷하지만, 버튼의 기능이 다릅니다. 기사님은 배차 취소(빨간색)와 배차 건의(주황색) 기능을 사용할 수 있습니다. JTable을 클릭하면 해당 주문 정보가 아래 텍스트 필드에 자동으로 채워지는 편의 기능도 똑같이 적용했습니다.

09

기사-배차내역(Search)/배차관리 (Update/Delete)

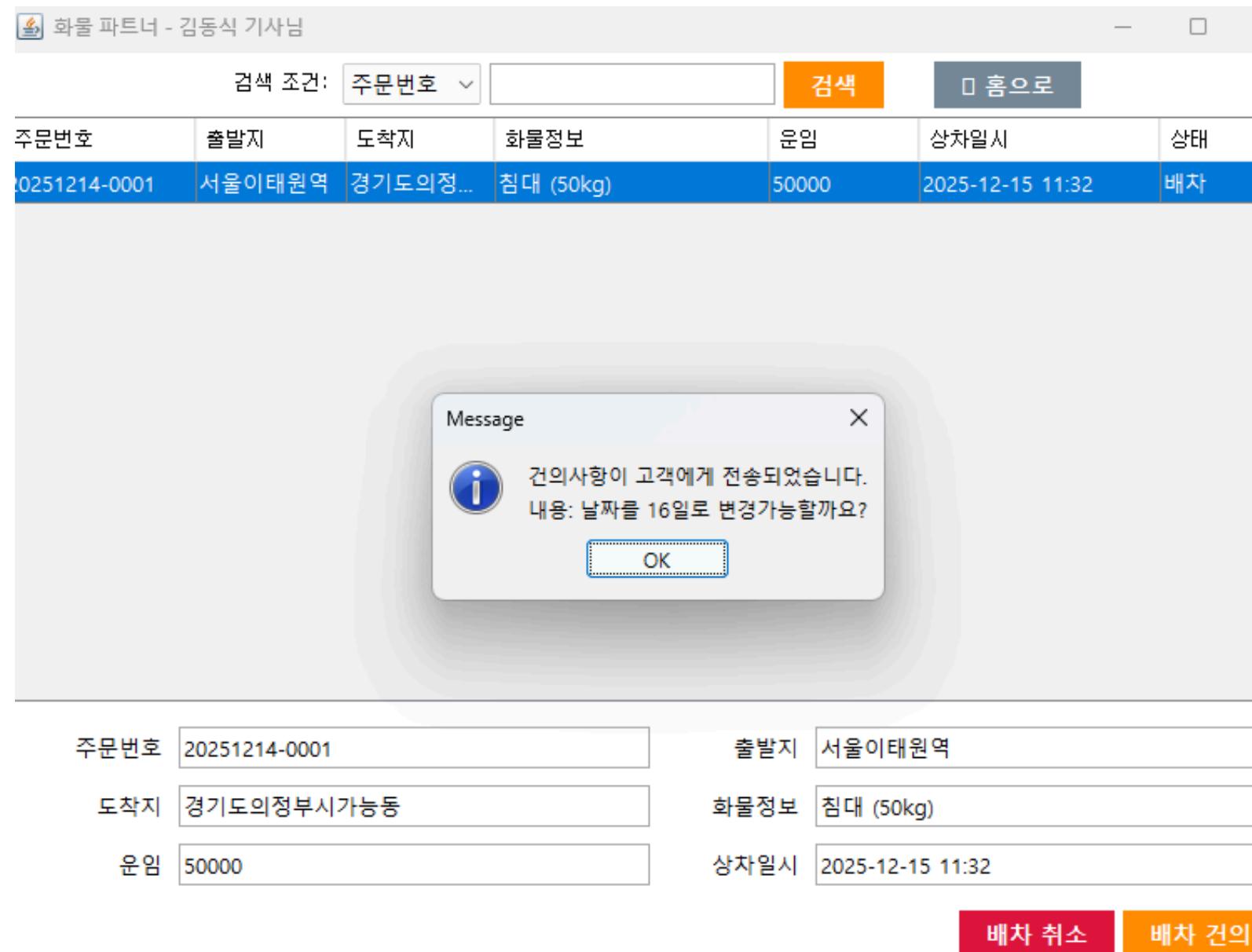


배차 건의 버튼을 눌렀을 때의 기능입니다. 고객과 직접 통화하기 곤란한 상황을 가정하여, 간단한 메시지를 보낼 수 있는 기능을 구현해 보았습니다.

JOptionPane.showInputDialog를 사용해서 텍스트를 입력받고, 확인을 누르면 전송되었다는 알림이 뜹니다. 실제로는 채팅 서버가 필요하겠지만, 이번 프로젝트에서는 메시지 전송 로직의 흐름을 구현하는 데에 의의를 두었습니다.

09

기사-배차내역(Search)/배차관리 (Update/Delete)



건의사항 입력을 마치고 확인을 누르면 뜨는 전송 완료 메시지입니다. 사용자가 버튼을 눌렀을 때 아무런 반응이 없으면 기능이 제대로 작동했는지 불안해할 수 있습니다. 그래서 JOptionPane.showMessageDialog를 사용하여 처리가 완료되었다는 피드백을 확실하게 주도록 구현했습니다. 또한 방금 입력한 내용을 메시지 창에 한 번 더 보여줌으로써, 사용자가 자신이 보낸 내용을 최종적으로 확인할 수 있게끔 UI 디테일을 챙겼습니다.

09

기사-배차내역(Search)/배차관리 (Update/Delete)



기사님이 배차받은 화물을 취소하려고 할 때 띄우는 확인 팝업입니다. 배차 취소는 기사님에게는 배차 내역 삭제지만, 시스템 전체로 보면 해당 주문이 다시 대기 상태로 돌아가야 하는 중요한 작업입니다. 그래서 실수로 누르는 것을 방지하기 위해 예/아니오 옵션을 주어 신중하게 결정하도록 유도했습니다.

09

기사-배차내역(Search)/배차관리 (Update/Delete)



배차가 정상적으로 취소되었을 때 사용자에게 보여주는 알림창입니다. 이 확인 버튼을 누르는 순간 내부적으로는 두 가지 쿼리가 실행됩니다. DISPATCH 테이블에서 해당 기사님의 배차 내역을 DELETE하고, 동시에 ORDER_SHEET 테이블의 상태값을 배차에서 다시 대기로 UPDATE 합니다. 이 로직을 통해 다른 기사님이 해당 주문을 다시 잡을 수 있게 됩니다.

09

기사-배차내역(Search)/배차관리 (Update/Delete)



배차 취소 후 갱신된 배차 관리 화면입니다. 방금 취소한 주문 건이 목록에서 즉시 사라진 것을 볼 수 있습니다. 데이터를 삭제한 후에는 반드시 리스트를 다시 조회(Select)해서 테이블 모델을 업데이트해줘야 데이터 불일치 문제가 생기지 않는다는 것을 이번 과제를 통해 확실히 배웠습니다.

10

기사-내정보수정(Update)

The screenshot shows a user profile update form titled "내 정보 수정". The form includes fields for basic information (아이디: dpem1234, 비밀번호: masked, 이름: 김동식, 전화번호: 01022392932) and vehicle information (차량 정보 수정 (기사) section: 차량번호: 28소8237, 차종: 카고, 적재량(kg): 5000). At the bottom are "취소" and "수정완료" buttons.

아이디	dpem1234
비밀번호	●●●●
이름	김동식
전화번호	01022392932
차량 정보 수정 (기사)	
차량번호	28소8237
차종	카고
적재량(kg)	5000

취소 수정완료

프로젝트의 마지막 기능인 기사님 정보 수정 화면입니다. 일반 고객과 다르게 기사님은 USERS 테이블의 개인정보와 VEHICLE 테이블의 차량 정보를 동시에 관리해야 합니다. 그래서 화면을 구성할 때 아래쪽에 차량 정보 패널을 따로 만들었습니다. 특히 차량번호는 기본키(PK) 역할을 하기 때문에 수정하지 못하도록 비활성화 처리했고, 차종이나 적재량 같은 변경 가능한 정보만 수정할 수 있게 구현했습니다. 두 테이블을 조인(JOIN)해서 데이터를 가져오는 쿼리를 짜느라 꽤 고생했지만 잘 작동해서 뿌듯했습니다.

End 마치며

이번 기말 과제를 수행하며 단순히 자바 문법을 아는 것과, 그것을 활용해 실제 작동하는 소프트웨어를 만드는 것은 천지 차이라는 것을 몸소 느꼈습니다.

초기 기획 단계에서는 "그냥 연결만 하면 되겠지"라고 단순하게 생각했지만, 실제 개발 과정에서 수많은 시행착오를 겪었습니다. 특히 한글 데이터 크기 문제로 발생한 ORA-12899 오류나, 기사님이 배차를 취소할 때 주문 상태를 다시 되돌려야 하는 로직 등을 해결하면서 DB 설계와 예외 처리의 중요성을 뼈저리게 배웠습니다.

비록 시간 관계상 실제 지도 API 연동이나 소켓 통신을 이용한 실시간 채팅까지는 구현하지 못해 아쉬움이 남지만, Java Swing을 이용한 직관적인 UI 구성과 JDBC를 활용한 안정적인 데이터 연동이라는 당초 목표는 120% 달성했다고 생각합니다.

강의실에서 배운 이론들이 내 손끝에서 실제 프로그램으로 만들어지는 과정은 정말 짜릿한 경험이었습니다. 이번 프로젝트를 통해 얻은 자신감을 바탕으로, 향후에는 웹 기반의 더 고도화된 물류 시스템에도 도전해보고 싶습니다.

THANK YOU

2501110200 김재영

교수님 1년동안 고생 많으셨습니다!