

# AIRLINE 데이터베이스 구현 및 Web 응용 - 최종 프로젝트 보고서

학번: 12201560

이름: 김미르

---

## 1. 개요

### 1.1 프로젝트 목표

이번 프로젝트는 AIRLINE 데이터베이스를 기반으로 항공기 예약 시스템을 설계하고 구현하는 것임. 기본적인 요구사항에 따라 FLIGHT, FLIGHT\_LEG, LEG\_INSTANCE, FARE, SEAT\_RESERVATION, AIRPLANE, AIRPLANE\_TYPE, CAN\_LAND, AIRPORT 등의 엔티티를 설계하고, 관리자 페이지와 사용자 페이지를 구현하는 것이 목표.

### 1.2 구현 기능

관리자 페이지:

- 공항 정보 입력/수정/삭제
- 항공기 정보 입력/수정/삭제
- 항공편 정보 입력/수정/삭제

사용자 페이지:

- 예약하기 기능 (트랜잭션 포함)
- 예약조회 기능
- 예약취소 기능 (트랜잭션 포함)

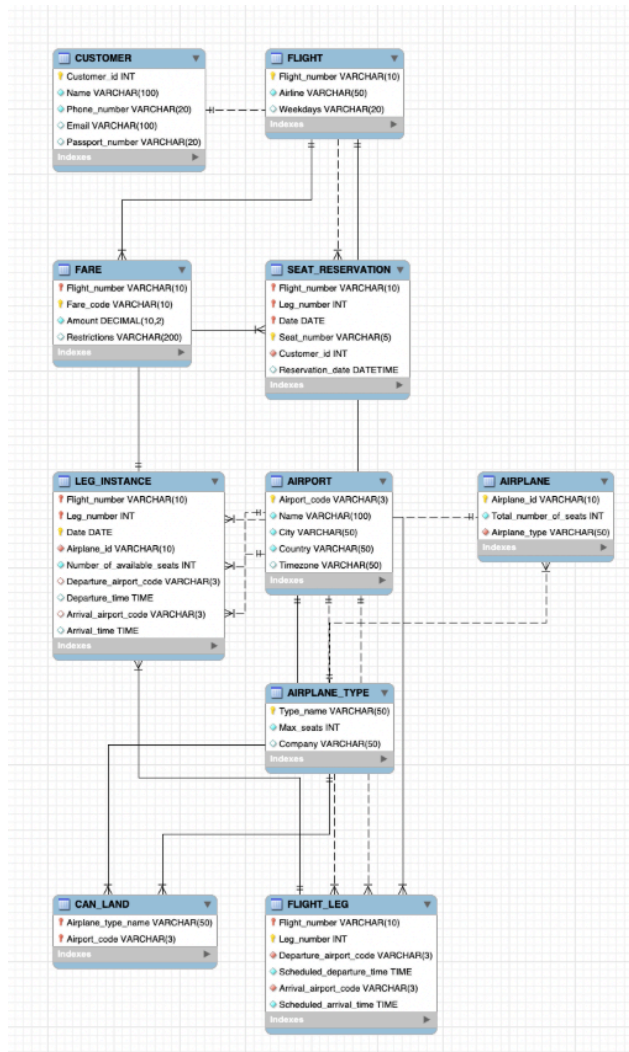
## 2. 상세 설계 내용

### 2.1 데이터베이스 스키마 설계

## 2.1.1 ERD 설계

EER Diagram:

MySQL Workbench에서 작성한 EER diagram은 아래와 같음.



엔티티 관계:

- AIRPORT ↔ FLIGHT\_LEG (1:N, 출발지/도착지)
- AIRPORT ↔ LEG\_INSTANCE (1:N, 출발지/도착지)
- AIRPORT ↔ AIRPLANE\_TYPE (N:M, CAN\_LAND를 통한 착륙 가능 관계)
- AIRPLANE\_TYPE → AIRPLANE (1:N)
- AIRPLANE → LEG\_INSTANCE (1:N)
- FLIGHT → FLIGHT\_LEG (1:N)
- FLIGHT → FARE (1:N)
- FLIGHT\_LEG → LEG\_INSTANCE (1:N)

- LEG\_INSTANCE → SEAT\_RESERVATION (1:N)
- CUSTOMER → SEAT\_RESERVATION (1:N)

데이터베이스는 다음과 같은 엔티티들로 구성되었음:

1. AIRPORT: 공항 정보 (Airport\_code, Name, City, Country, Timezone)
2. AIRPLANE\_TYPE: 항공기 타입 정보 (Type\_name, Max\_seats, Company)
3. AIRPLANE: 항공기 정보 (Airplane\_id, Total\_number\_of\_seats, Airplane\_type)
4. CAN\_LAND: 항공기 타입과 공항 간 착륙 가능 관계
5. FLIGHT: 항공편 정보 (Flight\_number, Airline, Weekdays)
6. FLIGHT\_LEG: 항공편 구간 정보 (Flight\_number, Leg\_number, Departure\_airport\_code, Scheduled\_departure\_time, Arrival\_airport\_code, Scheduled\_arrival\_time)
7. LEG\_INSTANCE: 항공편 구간 인스턴스 (날짜별) (Flight\_number, Leg\_number, Date, Airplane\_id, Number\_of\_available\_seats, ...)
8. FARE: 요금 정보 (Flight\_number, Fare\_code, Amount, Restrictions)
9. CUSTOMER: 고객 정보 (Customer\_id, Name, Phone\_number, Email, Passport\_number)
10. SEAT\_RESERVATION: 좌석 예약 정보 (Flight\_number, Leg\_number, Date, Seat\_number, Customer\_id, Reservation\_date)

### 2.1.2 정규화

모든 테이블은 3NF(제3정규형)까지 정규화되었음:

- 1NF: 모든 속성이 원자값을 가짐
- 2NF: 부분 함수 종속성 제거 (FLIGHT\_LEG의 복합키, LEG\_INSTANCE의 복합키 등)
- 3NF: 이행 함수 종속성 제거 (AIRPLANE의 Airplane\_type을 별도 테이블로 분리)

### 2.1.3 제약조건

#### Primary Key 제약조건:

- 각 테이블에 적절한 Primary Key 설정
- 복합키 사용 (FLIGHT\_LEG, LEG\_INSTANCE, SEAT\_RESERVATION 등)

#### Foreign Key 제약조건:

- 참조 무결성 보장
- ON DELETE CASCADE / ON DELETE RESTRICT 적절히 사용
- ON UPDATE CASCADE 사용

#### CHECK 제약조건:

- Leg\_number > 0
- Max\_seats > 0
- Total\_number\_of\_seats > 0
- Number\_of\_available\_seats >= 0
- Amount >= 0

#### 트리거를 통한 제약조건:

- 출발지와 도착지가 같을 수 없도록 검증 (FLIGHT\_LEG)
- LEG\_INSTANCE 삽입 시 좌석 수 자동 설정

#### 애플리케이션 레벨 제약조건:

- FLIGHT\_LEG 존재 여부 확인
- AIRPLANE 존재 여부 확인
- CAN\_LAND 관계 확인 (항공기 타입이 도착 공항에 착륙 가능한지)
- AIRPORT 존재 여부 확인
- 좌석 수 일치 확인

### 2.1.4 인덱스 설계

성능 최적화를 위해 다음과 같은 인덱스를 생성했음:

```

-- 기본 인덱스
CREATE INDEX idx_city ON AIRPORT(City);
CREATE INDEX idx_airplane_type ON AIRPLANE(Airplane_type);
CREATE INDEX idx_departure_airport ON FLIGHT_LEG(Departure_airport_code);
CREATE INDEX idx_arrival_airport ON FLIGHT_LEG(Arrival_airport_code);
CREATE INDEX idx_date ON LEG_INSTANCE(Date);
CREATE INDEX idx_airplane ON LEG_INSTANCE(Airplane_id);
CREATE INDEX idx_name ON CUSTOMER(Name);
CREATE INDEX idx_customer ON SEAT_RESERVATION(Customer_id);
CREATE INDEX idx_reservation_date ON SEAT_RESERVATION(Reservation_date);

-- 복합 인덱스
CREATE INDEX idx_flight_leg_composite ON FLIGHT_LEG(Flight_number, Leg_number);
CREATE INDEX idx_leg_instance_date_range ON LEG_INSTANCE(Date, Flight_number);
CREATE INDEX idx_seat_reservation_customer_date ON SEAT_RESERVATION(Customer_id, Date);

```

### 2.1.5 뷰 생성

복잡한 조인 쿼리를 뷰로 생성하여 재사용성을 높였음:

1. vw\_flight\_details: 항공편 상세 정보 (FLIGHT + FLIGHT\_LEG + AIRPORT 정보)
2. vw\_available\_flights: 예약 가능한 항공편 조회 (LEG\_INSTANCE + FLIGHT\_LEG + AIRPLANE + AIRPORT 정보)
3. vw\_customer\_reservations: 고객 예약 정보 (SEAT\_RESERVATION + CUSTOMER + FLIGHT\_LEG + FLIGHT + AIRPORT 정보)

### 2.1.6 트랜잭션 설계

예약하기 트랜잭션:

```

// setSeatReservation 함수에서 구현
await connection.beginTransaction();
// 1. 좌석 예약 삽입
INSERT INTO SEAT_RESERVATION ...
// 2. 사용 가능한 좌석 수 감소
UPDATE LEG_INSTANCE SET Number_of_available_seats = Number_of_available_seats - 1 ...
await connection.commit();
// 에러 발생 시 rollback

```

## 예약 취소 트랜잭션:

```
// deleteSeatReservation 함수에서 구현
await connection.beginTransaction();
// 1. 좌석 예약 삭제
DELETE FROM SEAT_RESERVATION ...
// 2. 사용 가능한 좌석 수 증가
UPDATE LEG_INSTANCE SET Number_of_available_seats = Number_of_available_seats + 1 ...
await connection.commit();
// 에러 발생 시 rollback
```

## 2.2 애플리케이션 설계

### 2.2.1 프로젝트 구조

final\_project/

- |— database/
  - | |— sql.js            데이터베이스 연결 및 SQL 쿼리 함수
- |— routes/
  - | |— admin.js        관리자 페이지 라우터
  - | |— user.js        사용자 페이지 라우터
- |— views/
  - | |— admin/        관리자 페이지 뷰
  - | |— user/        사용자 페이지 뷰
- |— public/
  - | |— css/
    - | |— style.css    스타일시트
- |— src/
  - | |— index.js        Express 서버 메인 파일
- |— database/
  - | |— create\_database.sql   데이터베이스 생성 스크립트

### 2.2.2 주요 구현 내용

데이터베이스 연결 (database/[sql.js](#)):

```
const pool = mysql.createPool({
  host: 'localhost',
  port: 3306,
  user: 'dbuser',
  password: 'c3409711!',
  database: 'AIRLINE',
  charset: 'utf8mb4',
  collation: 'utf8mb4_unicode_ci',
  connectionLimit: 10,
  acquireTimeout: 60000,
  timeout: 60000,
});
```

## 트랜잭션 구현 (database/sql.js - setSeatReservation):

```
setSeatReservation: async (data) => {
  const connection = await promisePool.getConnection();
  try {
    await connection.beginTransaction();

    // 좌석 예약 삽입
    const sql1 = `INSERT INTO SEAT_RESERVATION
      (Flight_number, Leg_number, Date, Seat_number, Customer_id)
      VALUES (?, ?, ?, ?, ?)`;
    await connection.query(sql1, [
      data.Flight_number,
      data.Leg_number,
      data.Date,
      data.Seat_number,
      data.Customer_id
    ]);

    // 사용 가능한 좌석 수 감소
    const sql2 = `UPDATE LEG_INSTANCE
      SET Number_of_available_seats = Number_of_available_seats - 1
      WHERE Flight_number = ? AND Leg_number = ? AND Date = ?`;
    await connection.query(sql2, [
      data.Flight_number,
      data.Leg_number,
      data.Date
    ]);

    await connection.commit();
  } catch (error) {
    await connection.rollback();
    throw error;
  } finally {
    connection.release();
  }
}
```

## 예약조건 검증 (database/sql.js - validateLegInstance):

```
validateLegInstance: async (data) => {
  const errors = [];

  // FLIGHT_LEG 존재 확인
  const flightLegExists = await validationSql.checkFlightLegExists(
    data.Flight_number, data.Leg_number
  );
  if (!flightLegExists) {
    errors.push('FLIGHT_LEG does not exist');
  }

  // AIRPLANE 존재 확인
  const airplaneExists = await validationSql.checkAirplaneExists(data.Airplane_id);
  if (!airplaneExists) {
    errors.push('AIRPLANE does not exist');
  }

  // CAN_LAND 관계 확인
  const canLand = await validationSql.checkCanLandRelation(
    data.Airplane_id, data.Arrival_airport_code
  );
  if (!canLand) {
    errors.push('Airplane type cannot land at arrival airport');
  }

  return {
    valid: errors.length === 0,
    errors: errors
  };
}
```

## 라우팅 구조:

- /admin: 관리자 메인 페이지
- /admin/airport: 공항 관리
- /admin/airplane: 항공기 관리
- /admin/flight: 항공편 관리
- /user: 사용자 메인 페이지 (항공편 검색)
- /user/search: 검색 결과
- /user/reserve/:flightNumber/:legNumber/:date: 예약하기
- /user/reservation: 예약 조회
- /user/cancel: 예약 취소

---

## 3. 실행 화면

### 3.1 사용자 페이지



### 3.1.1 항공편 검색 화면

AIRLINE Reservation System

[홈](#)[관리자](#)[사용자](#)

AIRLINE Reservation System

[홈](#)[관리자](#)[사용자](#)

항공편 검색

출발지:  
선택하세요

도착지:  
선택하세요

출발일:  
2025. 11. 16.

검색

[예약 조회](#)

© 2025 AIRLINE Reservation System

© 2025 AIRLINE Reservation System

항공편 검색을 위한 출발지, 도착지, 출발일을 선택할 수 있는 화면임.

### 3.1.2 검색 결과 화면

AIRLINE Reservation System

[홈](#)[관리자](#)[사용자](#)

AIRLINE Reservation System

[홈](#)[관리자](#)[사용자](#)

검색 결과

출발지: ICN | 도착지: NRT | 날짜: 2025-11-16

항공사	출발지	출발시간	도착지	도착시간	사용가능 좌석	원차기	예약
KE001	Incheon International Airport (ICN)	08:00:00	Narita International Airport (NRT)	10:30:00	180	500000.00 원	<a href="#">예약하기</a>

[다시 검색](#)

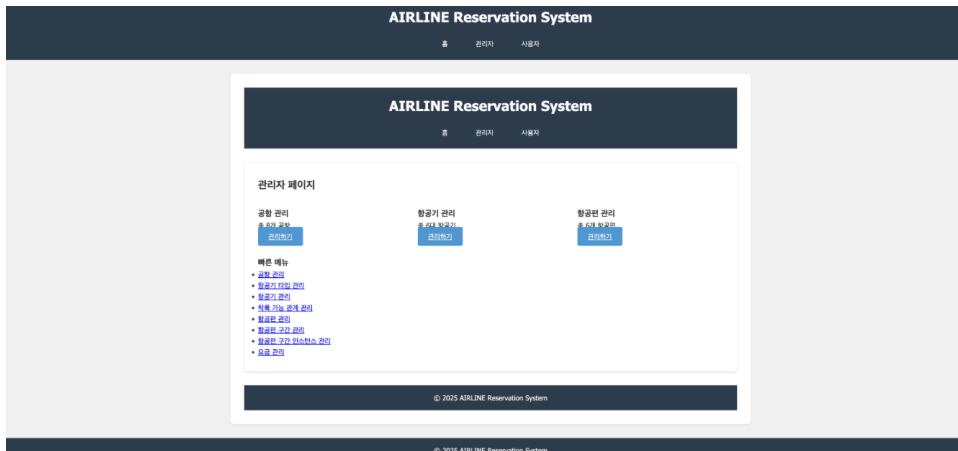
© 2025 AIRLINE Reservation System

© 2025 AIRLINE Reservation System

검색 조건에 맞는 항공편 목록이 표시되고, 각 항공편에 대해 예약하기 버튼이 제공됨.

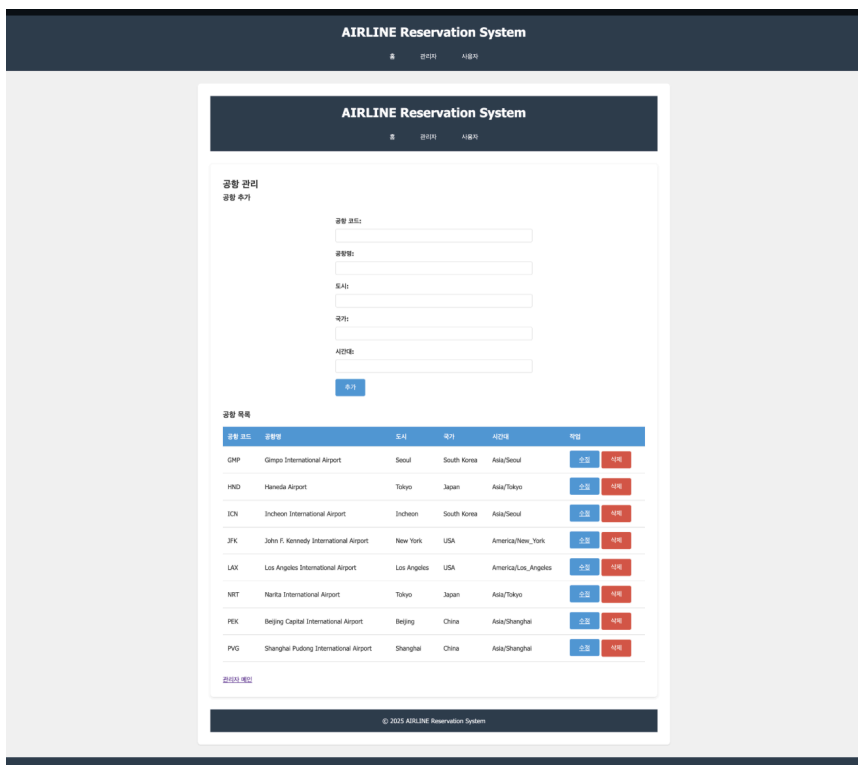
## 3.2 관리자 페이지

### 3.2.1 관리자 메인 화면



관리자 페이지의 메인 화면으로, 공항, 항공기, 항공편의 개수를 표시하고 각 관리 메뉴로 이동할 수 있음.

### 3.2.2 공항 관리 화면



공항 정보를 추가, 수정, 삭제할 수 있는 화면임. 공항 목록이 테이블 형태로 표시되고, 각 공항에 대해 수정/삭제 버튼이 제공됨.

### 3.2.3 항공기 관리 화면

AIRLINE Reservation System

[홈](#)
[관리자](#)
[사용자](#)

AIRLINE Reservation System

[홈](#)
[관리자](#)
[사용자](#)

항공기 관리

항공기 추가

항공기 ID:

좌석 수:

항공기 타입:

Airbus A320

추가

항공기 목록

ID	좌석수	타입	작업
KR001	180	Boeing 737	<a href="#">수정</a> <a href="#">삭제</a>
KR002	180	Boeing 737	<a href="#">수정</a> <a href="#">삭제</a>
KR003	350	Boeing 777	<a href="#">수정</a> <a href="#">삭제</a>
KR004	180	Airbus A320	<a href="#">수정</a> <a href="#">삭제</a>
KR005	555	Airbus A380	<a href="#">수정</a> <a href="#">삭제</a>
KR006	290	Boeing 787	<a href="#">수정</a> <a href="#">삭제</a>

[관리자 메뉴](#)

© 2025 AIRLINE Reservation System

항공기 정보를 추가, 수정, 삭제할 수 있는 화면임. 항공기 목록이 테이블 형태로 표시되고, 항공기 타입을 선택하여 항공기를 추가할 수 있음.

### 3.2.4 항공편 관리 화면

AIRLINE Reservation System

[홈](#)
[관리자](#)
[사용자](#)

AIRLINE Reservation System

[홈](#)
[관리자](#)
[사용자](#)

항공편 관리

항공편 추가

항공편 ID:

출발지:

도착지:

항공편 타입:

Boeing 737-800

추가

항공편 목록

ID	출발지	도착지	항공편 타입	작업
KR001	Seoul	Paris	Boeing 737-800	<a href="#">수정</a> <a href="#">삭제</a>
KR002	Seoul	Paris	Boeing 737-800	<a href="#">수정</a> <a href="#">삭제</a>
KR003	Seoul	Paris	Boeing 737-800	<a href="#">수정</a> <a href="#">삭제</a>
KR004	Seoul	Paris	Boeing 737-800	<a href="#">수정</a> <a href="#">삭제</a>
KR005	Seoul	Paris	Boeing 737-800	<a href="#">수정</a> <a href="#">삭제</a>
KR006	Seoul	Paris	Boeing 737-800	<a href="#">수정</a> <a href="#">삭제</a>

[관리자 메뉴](#)

© 2025 AIRLINE Reservation System

항공편 정보를 추가, 수정, 삭제할 수 있는 화면임. 항공편 목록이 테이블 형태로 표시되고, 각 항공편에 대해 수정/삭제 버튼이 제공됨.

### 3.3 사용자 페이지 (예약 기능)

#### 3.3.1 예약하기 화면

AIRLINE Reservation System

[홈](#)[관리자](#)[사용자](#)

예약하기

항공편 정보

항공편: KE101 | 구간: 1  
출발: Gimpo International Airport (GMP) - 09:00:00  
도착: Haneda Airport (HND) - 11:00:00  
날짜: Mon Nov 17 2025 00:00:00 GMT+0900 (Korean Standard Time)  
좌석 선택:  

선택하세요

고객 정보

이름:  
  
전화번호:  
  
이메일:  
  
여권번호:

예약하기

취소

© 2025 AIRLINE Reservation System

항공편을 선택한 후 좌석과 고객 정보를 입력하여 예약할 수 있는 화면임.

#### 3.3.2 예약 조회 화면

AIRLINE Reservation System

[홈](#)[관리자](#)[사용자](#)

예약 조회

전화번호:

010-1234-5678

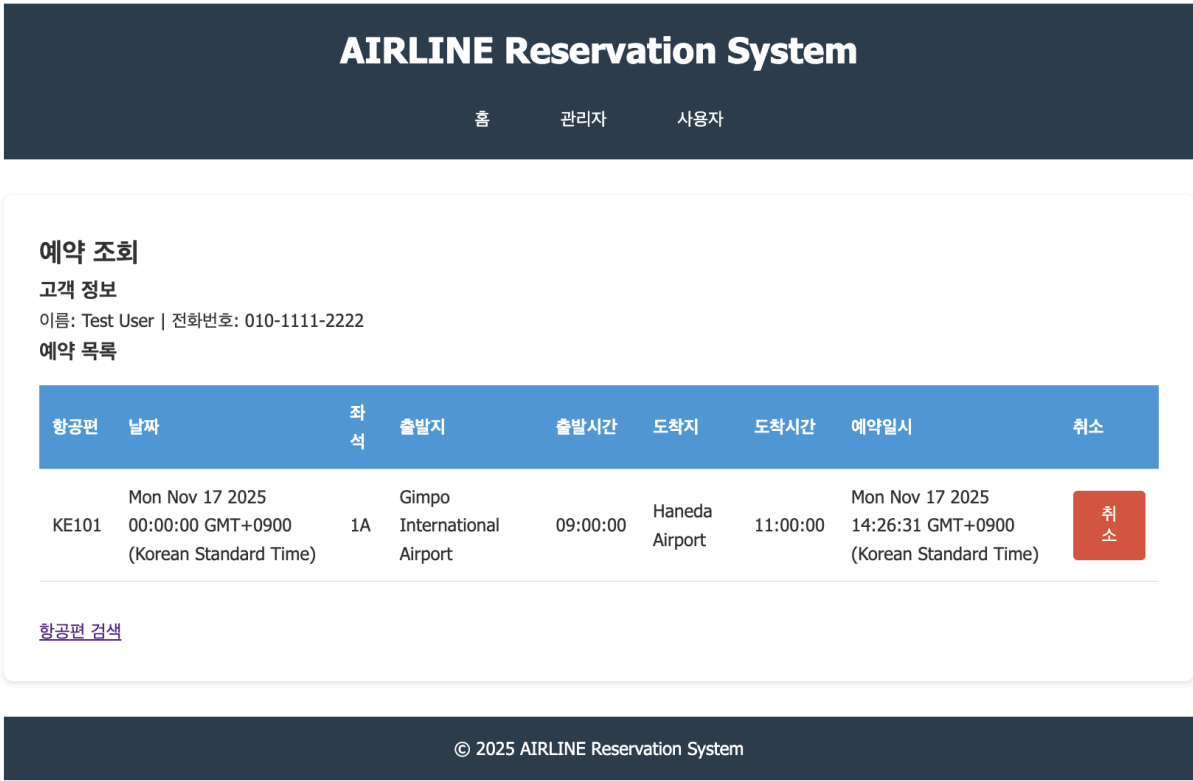
조회

[항공편 검색](#)

© 2025 AIRLINE Reservation System

전화번호를 입력하여 예약 정보를 조회할 수 있는 화면임.

### 3.3.3 예약 목록 화면



조회한 고객의 예약 목록을 확인할 수 있는 화면임. 각 예약에 대해 취소 버튼이 제공됨.

---

## 4. 고찰

### 4.1 구현 성공 사항

1. 데이터베이스 설계: 요구사항에 맞는 모든 엔티티와 관계를 설계하고, 정규화를 통해 데이터 중복을 최소화했음.

2. 제약조건 구현: Primary Key, Foreign Key, CHECK 제약조건, 트리거, 애플리케이션 레벨 검증을 통해 데이터 무결성을 보장했음.

3. 성능 최적화: 자주 조회되는 컬럼에 인덱스를 생성하고, 복합 인덱스를 통해 쿼리 성능을 향상시켰음.

4. 뷰 활용: 복잡한 조인 쿼리를 뷰로 생성하여 코드 재사용성을 높이고 유지보수를 용이하게 했음.

5. 트랜잭션 구현: 예약하기와 예약 취소 기능에서 트랜잭션을 사용하여 데이터 일관성을 보장했음.

6. 사용자 인터페이스: 관리자와 사용자 페이지를 분리하여 각각의 역할에 맞는 인터페이스를 제공했음.