

# 웹 시스템 설계 HW#2

추민솔 TA

[cmss1217@ajou.ac.kr](mailto:cmss1217@ajou.ac.kr)

2025-10-17

## 목차

<b>1 과제</b>	<b>2</b>
1.1 목표 . . . . .	2
1.2 과제 개요 . . . . .	2
1.3 과제 코드 구성 . . . . .	2
1.4 구현 요구사항 . . . . .	2
1.4.1 서버 구현 ( <code>server.js</code> ) . . . . .	2
1.4.2 클라이언트 구현 ( <code>index.html, style.css</code> ) . . . . .	3
1.5 예시 동작 흐름 . . . . .	4
1.6 스켈레톤 코드 공통 설정 설명 . . . . .	4
1.7 제출 요구사항 . . . . .	4
<b>2 Appendix</b>	<b>5</b>

# 1 과제

## 1.1 목표

- HW#1에서 구현한 REST API 기반 비행기 좌석 예약 코드를 발전시켜, 파일 기반 데이터 저장과 예약 삭제 기능을 추가한다.
- Node.js + Express를 활용해 CRUD 동작 중 **DELETE** 처리를 익히고, 파일 시스템 모듈(fs)을 사용해 데이터 지속성(Persistence)을 구현한다.
- 클라이언트 측에서 비동기 요청(fetch)을 통해 예약 삭제를 포함한 전체 기능을 제어하도록 구성한다.

## 1.2 과제 개요

- 본 과제는 **HW#1**의 확장 버전으로, 서버가 데이터를 파일(.json)로 관리하며 간단한 CRUD 중 일부 기능을 완성하는 것이다.
- 사용자는 웹 페이지를 통해 항공편 정보 확인, 예약 등록 및 삭제, 예약 목록 조회를 수행할 수 있다.
- 서버는 두 .json 파일을 통해 정보를 읽고, 수정될 경우 파일에 반영한다.

## 1.3 과제 코드 구성

- 본 과제는 서버(백엔드)와 클라이언트(프론트엔드)로 구성된다.
  - `server.js` Express 기반 REST API 서버. 항공편/예약 정보 관리 및 파일 입출력.
  - `index.html` 클라이언트 페이지. 비동기 fetch 기반으로 예약 확인·생성·삭제 요청 처리.
  - `style.css` 스타일시트. 별도 파일로 작성하여 `index.html`에서 `<link>`로 포함 가능.
  - `data/flight.json`, `data/reservations.json` 서버가 참조 및 수정하는 데이터 파일.

## 1.4 구현 요구사항

### 1.4.1 서버 구현 (`server.js`)

서버는 항공편 및 예약 정보를 관리하며, 파일을 통해 데이터를 저장하고 불러온다. 그 과정에서 REST API를 제공하여 예약 조회, 생성, 삭제 요청을 처리하며, 모든 변경 사항을 .json 파일에 반영한다.

**필수 엔드포인트** 기존 HW#1의 구현 요건이었던 3개의 API endpoint는 HW#1과 동일한 로직으로 처리된다. 다만, 이번 과제에서는 삭제 기능이 추가되었으며, 항공편 정보 및 예약 목록을 .json 파일 기반으로 관리한다.

- GET /api/flight

항공편 정보를 JSON으로 반환하며, 필수 필드는 아래와 같다.

(구현 방식은 HW#1과 동일하되, 이번에는 `data/flight.json` 파일에서 로드한다.)

- `code`, `from`, `to`, `date`, `departureTime`, `arrivalTime`
- `rows`: 총 행 수 (예: 12), `cols`: 사용 가능한 열 배열 (예: `["A", "B", "C", "D", "E", "F"]`)

- GET /api/reservations

현재 예약 목록을 JSON 배열로 반환한다. 각 예약 객체의 예시는 { id, name, seat, ts } 형태이다.  
(HW#1의 메모리 배열 대신 data/reservations.json 파일 내용을 사용한다.)

- POST /api/reservations

요청 본문(name, seat)을 검증 후 새로운 예약을 생성한다.  
(HW#1 의 메모리 배열 대신, data/reservations.json 파일에 반영한다.)

- name: 앞뒤 공백을 제거한 길이가 2자 이상이어야 한다. (미만이면 400 Bad Request)
- seat: 형식은 <행번호><열문자> (예: 1A, 12F). (불일치 시 400 Bad Request)
- 좌석은 항공편 범위 내여야 한다.
- 이미 예약된 좌석이면 409 Conflict를 통해 요청을 거부한다.
- 성공 시 201 Created와 생성된 객체({ id, name, seat, ts })를 반환하며, data/reservations.json 파일에 반영한다.

- DELETE /api/reservations/:id

경로 매개변수 id로 전달된 예약을 삭제하며, data/reservations.json 파일에 반영한다.

- 존재하지 않으면 404 Not Found와 메시지 "해당 예약이 없습니다." 반환.
- 성공 시 200 OK와 함께 삭제된 객체 정보({ id, name, seat, ts }) 반환 및 data/reservations.json 파일에 반영.

## 필수 요소

- 예약 객체의 id 값은 중복되면 안되며, 삭제 시에도 재사용되어서는 안된다. (Appendix 삭제 예시 참고)
- 항공편 정보 및 예약 목록은 파일 기반으로 관리하며, 서버 재시작 시 유지되어야 한다.

### 1.4.2 클라이언트 구현 (index.html, style.css)

클라이언트는 비동기 요청(fetch)을 통해 서버와 통신하며, 항공편 정보 조회, 예약 등록 및 삭제 기능을 제공한다. 또한 style.css를 이용해 전체 페이지가 단순하고, 직관적으로 보이도록 스타일을 적용한다.

## 필수 요소

- 탑승객 이름 입력 필드 (name)와 좌석 입력 필드 (seat) 검증 로직이 존재해야 한다.
- 예약 관련 요청은 모두 비동기 방식(fetch)으로 처리해야 한다.
  - 예약 생성 폼 — POST /api/reservations 호출
  - 예약 목록 테이블 — GET /api/reservations 결과 표시
  - 각 예약 행의 삭제 버튼 — 사용자 의사 재확인 후 DELETE /api/reservations/:id 호출
  - 항공편 정보 상단 표시 — GET /api/flight 호출 결과 활용

- 매 요청의 결과는 즉각적으로 반영되어야 한다.  
(Ex. 삭제 시 바로 클라이언트의 예약 목록 테이블에서 제거, 예약 시 바로 예약 목록에서 확인 가능 등)
- 상태 바를 추가하여 모든 요청에 대한 결과를 사용자가 직관적으로 파악할 수 있게 한다.

## 1.5 예시 동작 흐름

- 메인 페이지(index.html) 접속 시, GET /api/flight 및 GET /api/reservations 호출을 통해 항공편 정보 및 예약 목록을 별도의 동작 없이 확인할 수 있다.
- 이름 및 좌석 입력 후 예약 버튼 클릭 시 → POST /api/reservations로 요청을 보내며, 성공 시 상태 바에 표기하며 예약 목록이 변경된다.
- 각 행의 삭제 버튼 클릭 시 → DELETE /api/reservations/:id로 요청을 보내며, 성공 시 상태 바에 표기하며, 예약 목록에 반영한다.
- 서버 재시작 시, 기존의 항공편 정보와, 예약 목록을 그대로 확인할 수 있다.

## 1.6 스켈레톤 코드 공통 설정 설명

과제와 함께 제공되는 `server.js` 스켈레톤 코드에는 구현의 편의성을 위해 몇 가지 공통 설정이 포함되어있다. 특히나 이번 과제에서는, HW#1에 더해, `fs` 모듈이 추가되었으며, 간단한 설명은 아래와 같다.

### fs 모듈

- Node.js 내장 파일 시스템 모듈.
- `readFileSync`, `writeFileSync`을 이용하여 동기식으로 파일을 읽고 쓸 수 있다.
- `readFile`, `writeFile`을 사용하면 비동기적으로 파일 입출력을 처리할 수 있다.

## 1.7 제출 요구사항

- HW2\_학번\_이름.zip 형식으로 압축하여 제출

### 예시

```
HW2_202301234_홍길동.zip/
├── server.js
├── index.html
├── style.css
└── data/
    ├── flight.json
    └── reservations.json
```

- 제출 기한: 10/29 (수) 23:59 (지각 제출 시 D+1 -40%, D+2 -70%, D+3 -100% 감점)

## 2 Appendix

### 비행기 좌석 예약

The screenshot shows a flight seat reservation interface. At the top, a summary box displays: AJ101 | PAL → HYK | 2025-10-14 | 10:30 → 11:40. Below this, the '좌석 예약' (Seat Reservation) section contains fields for '탑승객 이름' (Passenger Name) with '홍길동' entered, and '좌석' (Seat) with '예: 3C' selected. A note below the seat selection field specifies: 예약할 좌석을 입력하세요. 형식: 행번호 + 열(예: 7B). 입력 가능 범위: 행 1-12 / 열 A, B, C, D, E, F. A dark blue '예약하기' (Book) button is at the bottom. To the right, a '예약 목록' (Reservation List) table is shown with columns: ID, 이름, 좌석, 시간, and 작업. It displays the message: 예약이 없습니다.

ID	이름	좌석	시간	작업
예약이 없습니다.				

Figure 1: 초기 화면 예시  
빨간 네모 상자와 같이 항공편 정보 표기

## 비행기 좌석 예약

AJ101 | PAL → HYK | 2025-10-14 | 10:30 → 11:40

좌석 예약		예약 목록			
ID	이름	좌석	시간	작업	
예약이 없습니다.					

**탑승객 이름**

2글자 이상 입력하세요.

**좌석**

예약할 좌석을 입력하세요. 형식: 행번호 + 열(예: 7B)  
입력 가능 범위: 행 1-12 / 열 A, B, C, D, E, F

**예약하기**

**이름을 2자 이상 입력하세요.**

(a) 빈 이름 처리 예시

## 비행기 좌석 예약

AJ101 | PAL → HYK | 2025-10-14 | 10:30 → 11:40

좌석 예약		예약 목록			
ID	이름	좌석	시간	작업	
예약이 없습니다.					

**탑승객 이름**

2글자 이상 입력하세요.

**좌석**

예약할 좌석을 입력하세요. 형식: 행번호 + 열(예: 7B)  
입력 가능 범위: 행 1-12 / 열 A, B, C, D, E, F

**예약하기**

**좌석 형식이 올바르지 않습니다. 예: 3C / 12A**

(b) 빈 좌석 처리 예시

Figure 2: 비어있는 값 처리 예시 (a) 이름 (b) 좌석

## 비행기 좌석 예약

AJ101 | PAL → HYK | 2025-10-14 | 10:30 → 11:40

좌석 예약					예약 목록														
ID	이름	좌석	시간	작업	ID	이름	좌석	시간	작업										
1	홍길동	1A	2025. 10. 14. 오전 1:15:25	<button>삭제</button>	1	홍길동	1A	2025. 10. 14. 오전 1:15:25	<button>삭제</button>										
<input type="text" value="홍길동"/> 2글자 이상 입력하세요.					<table border="1"> <thead> <tr> <th>ID</th> <th>이름</th> <th>좌석</th> <th>시간</th> <th>작업</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>홍길동</td> <td>1A</td> <td>2025. 10. 14. 오전 1:15:25</td> <td><button>삭제</button></td> </tr> </tbody> </table>					ID	이름	좌석	시간	작업	1	홍길동	1A	2025. 10. 14. 오전 1:15:25	<button>삭제</button>
ID	이름	좌석	시간	작업															
1	홍길동	1A	2025. 10. 14. 오전 1:15:25	<button>삭제</button>															
<input type="text" value="3C"/> 예약할 좌석을 입력하세요. 형식: 행번호 + 열(0이 7B) 입력 가능한 범위: 행 1-12 / 열 A, B, C, D, E, F					예약 완료: 홍길동 / 1A														
<input type="button" value="예약하기"/>																			

(a) 예약 등록 화면 예시

## 비행기 좌석 예약

AJ101 | PAL → HYK | 2025-10-14 | 10:30 → 11:40

좌석 예약					예약 목록														
ID	이름	좌석	시간	작업	ID	이름	좌석	시간	작업										
1	홍길동	1A	2025. 10. 14. 오전 1:15:25	<button>삭제</button>	1	홍길동	1A	2025. 10. 14. 오전 1:15:25	<button>삭제</button>										
<input type="text" value="김민수"/> 2글자 이상 입력하세요.					<table border="1"> <thead> <tr> <th>ID</th> <th>이름</th> <th>좌석</th> <th>시간</th> <th>작업</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>홍길동</td> <td>1A</td> <td>2025. 10. 14. 오전 1:15:25</td> <td><button>삭제</button></td> </tr> </tbody> </table>					ID	이름	좌석	시간	작업	1	홍길동	1A	2025. 10. 14. 오전 1:15:25	<button>삭제</button>
ID	이름	좌석	시간	작업															
1	홍길동	1A	2025. 10. 14. 오전 1:15:25	<button>삭제</button>															
<input type="text" value="1A"/> 예약할 좌석을 입력하세요. 형식: 행번호 + 열(0이 7B) 입력 가능한 범위: 행 1-12 / 열 A, B, C, D, E, F					이미 예약된 좌석입니다: 1A														
<input type="button" value="예약하기"/>																			

(b) 중복 좌석 처리 예시

Figure 3: 예약 등록 및 중복 처리 전체 예시 (a) 예약 등록 (b) 중복 처리

## 비행기 좌석 예약

AJ101 | PAL → HYK | 2025-10-14 | 10:30 → 11:40

좌석 예약					예약 목록				
ID	이름	좌석	시간	작업	ID	이름	좌석	시간	작업
1	홍길동	1A	2025. 10. 14. 오전 1:15:25	<button>삭제</button>					

탑승객 이름  
김민수  
2글자 이상 입력하세요.

좌석  
1AB  
예약할 좌석을 입력하세요. 형식: 행번호 + 열(0이: 7B)  
입력 가능 범위: 행 1-12 / 열 A, B, C, D, E, F

예약하기

좌석 형식이 올바르지 않습니다. 예: 3C / 12A

(a) 잘못된 좌석 번호 입력 예시

## 비행기 좌석 예약

AJ101 | PAL → HYK | 2025-10-14 | 10:30 → 11:40

좌석 예약					예약 목록				
ID	이름	좌석	시간	작업	ID	이름	좌석	시간	작업
1	홍길동	1A	2025. 10. 14. 오전 1:15:25	<button>삭제</button>					

탑승객 이름  
김민수  
2글자 이상 입력하세요.

좌석  
13A  
예약할 좌석을 입력하세요. 형식: 행번호 + 열(0이: 7B)  
입력 가능 범위: 행 1-12 / 열 A, B, C, D, E, F

예약하기

좌석 범위를 벗어났습니다. 유효 범위: 행 1-12, 열 A, B, C, D, E, F

(b) 범위를 벗어난 입력 예시

Figure 4: 잘못된 입력 처리 전체 예시 (a) 잘못된 좌석 번호 (b) 범위 초과 입력

The screenshot shows a meal reservation interface. On the left, there's a form for a meal reservation with fields for 'Passenger Name' (Hong Gil Dong) and 'Meal' (3C). On the right, there's a table of reservations. A modal dialog box is overlaid on the screen, containing the text 'localhost:3000 내용: #2 예약을 삭제할까요?' (localhost:3000 content: #2 Do you want to delete the reservation?) with '취소' (Cancel) and '확인' (Confirm) buttons.

ID	이름	좌석	시간	작업
1	홍길동	1A	2025. 10. 14. 오전 1:15:25	<button>삭제</button>
2	김민수	12D	2025. 10. 14. 오전 1:16:40	<button>삭제</button>

(a) 삭제 확인 창

## 비행기 좌석 예약

This screenshot shows the same meal reservation interface as above, but after a deletion. The modal dialog has been replaced by a green success message box at the bottom stating '예약 삭제 완료: 김민수 / 12D' (Reservation deletion completed: Kim Min Soo / 12D).

ID	이름	좌석	시간	작업
1	홍길동	1A	2025. 10. 14. 오전 1:15:25	<button>삭제</button>

(b) 삭제 완료 화면

Figure 5: 삭제 과정 예시 (a) 삭제 확인 (b) 삭제 완료

## 비행기 좌석 예약

AJ101 | PAL → HYK | 2025-10-14 | 10:30 → 11:40

ID	이름	좌석	시간	작업
1	홍길동	1A	2025. 10. 14. 오전 1:15:25	<button>삭제</button>
3	김민지	7C	2025. 10. 14. 오전 1:17:19	<button>삭제</button>

**좌석 예약**

탑승객 이름

2글자 이상 입력하세요.

좌석

예약할 좌석을 입력하세요. 형식: 행번호 + 열(예: 7B)  
입력 가능 범위: 행 1-12 / 열 A, B, C, D, E, F

**예약하기**

예약 완료: 김민지 / 7C

Figure 6: 추가 예약 예시