웹 시스템 설계 HW#3

추민솔 TA
cmss1217@ajou.ac.kr

2025-11-11

목차

1 과제

1.1 목표

- 기존 HW#2를 **React 컴포넌트 기반 SPA**로 재구성하고, `useState`와 `setter`로 상태를 정의 및 갱신하는 패턴을 익힌다.
- **타이머 기반 UI**(삭제 유예 / 취소 등)를 통해 상태 변화에 따른 UI 업데이트를 익힌다.

1.2 과제 개요

- 본 과제는 **HW#2**의 확장 버전으로, 기존 코드를 **React 컴포넌트 기반 SPA**로 전환하여 동일한 좌석 예약 기능을 구성한다.
- 사용자는 웹 페이지를 통해 기존과 같은 정보 확인, 예약 등록 등을 수행하며, **타이머 기반 UI(삭제 유예)**를 통해 상태 변화를 직관적으로 확인한다.
- 애플리케이션은 `App.jsx`에서 상태를 관리하고, 하위 컴포넌트로 전달하며, **fetch**로 기존 REST API와 통신한다.

1.3 과제 준비

과제를 진행하기 전, 아래 명령어를 실행한다. 아래 명령어를 순서대로 실행하면, `./frontend`에 React+Vite 프로젝트가 생성되고, 의존성 설치 후 Vite 개발 서버가 기동된다. 이후 브라우저에서 터미널에 표시된 localhost로 접속한다면 기본 화면을 확인할 수 있다.

```
$ npm create vite@latest frontend -- --template react
$ cd frontend
$ npm install
$ npm run dev
```

위 명령어를 실행한 후 확인되는 주요 파일에 대한 설명은 아래와 같다.

- `index.html` 클라이언트 앱이 탑재되는 단일 HTML 셸.
- `eslint.config.js` 코드 품질 검사 도구인 ESLint의 설정 파일.
- `vite.config.js` 빌드/개발 서버 동작을 정의하는 Vite 설정.
- `public/` 정적 자원(이미지, 아이콘 등)을 보관하는 폴더.
- `src/main.jsx` 애플리케이션 진입점(React 마운트 시작 지점).
- `src/App.jsx` 최상위 UI 컴포넌트(페이지 구성과 상태 전달의 중심).

1.4 과제 코드 구성

본 과제는 서버(Backend)와 클라이언트(Frontend)로 구성된다.

- Backend (기존 HW#2와 동일)
  - `server.js` Express 기반 REST API 서버. 항공편/예약 정보 관리 및 파일 입출력.
  - `data/flight.json`, `data/reservations.json` 서버가 참조 및 수정하는 데이터 파일.
- Frontend
  - `src/App.jsx` 클라이언트 최상위 페이지. 상위 컴포넌트로써 하위 컴포넌트들에게 상태를 전달하며, 비동기 fetch 기반으로 비행 조회, 예약 조회, 생성, 삭제 요청들을 처리.
  - 하위 컴포넌트.
    - `src/components/FlightMeta.jsx` 비행편 메타 정보(편명, 시간 등) 표시.
    - `src/components/ReservationFrom.jsx` 이름/좌석 입력, 형식/범위 검증, 제출 및 로딩 상태 관리.
    - `src/components/ReservationTable.jsx` 예약 목록 렌더링, 지연 삭제 및 삭제 취소 처리.
    - `src/index.css` or `src/App.css` 스타일시트. 전역 및 컴포넌트 스타일 정의.

1.5 구현 요구사항

본 과제에서는 Frontend 구현이 요구되며, Backend의 경우는 HW#2에서 구현된 코드를 그대로 사용한다.

1.5.1 클라이언트 구현 (Frontend)

클라이언트는 항공편 예약 데이터 loading과 예약 생성, 예약 목록 표시, 지연 삭제 및 삭제 취소, 상태 알림 표시까지 전체 상호작용을 담당한다.

각 항목 별 구현 사항

- 아래 항목들에서 지칭되는 '상위' 는 `App.jsx`를 의미한다.
- `src/main.jsx`
  - 애플리케이션 진입점으로서 단일 `<App/>` 컴포넌트를 루트 DOM에 마운트한다.
  - 전역 스타일/설정이 있다면 이 지점에서 한 번만 적용한다.
- `src/App.jsx`
  - 초기화 시점에 서버에 `GET /api/flight`, `GET /api/reservations`를 호출해 비행편 정보와 예약 목록을 로드하고, 로딩/에러 상태를 관리한다.
  - 전역 상태(`useState`)를 보유하고 하위 컴포넌트에 내려준다.
  - form 입력 변경을 제어(Controlled Inputs)하며, 제출 시 기본 검증을 수행한다. 이름 길이(최소 2자), 좌석 형식 및 범위.

  - 예약 생성 흐름을 처리한다. `POST /api/reservations`로 서버에 전송하고, 성공하면 목록을 다시 불러와 form을 초기화한다.
  - 삭제 지연 및 취소를 관리한다. 삭제 버튼 클릭 시, 카운트다운을 시작하며 1) 카운트다운 만료 시 `DELETE /api/reservations/:id` 호출, 2) 카운트다운 만료 전 취소 버튼 클릭 시 타이머를 해제시키고, 삭제를 진행하지 않는다.
  - 예약 성공, 예약 삭제, 에러 등을 상태 바로 관리하며, 일정 시간 이후 상태 바를 숨긴다.

- `src/components/FlightMeta.jsx`
  - 상위에서 전달받은 비행편 메타(`code`, 출발지/도착지, 날짜, 출/도착 시간)를 한 줄 요약 형태로 표시한다.

- `src/components/ReservationForm.jsx`
  - 이름/좌석 입력 필드를 제어 컴포넌트로 제공하고, 입력 변경을 상위 콜백으로 즉시 전달한다.
  - 제출 버튼 클릭 시 상위에 예약 생성을 요청한다.

- `src/components/ReservationTable.jsx`
  - 예약 목록을 표로 렌더링한다.
  - 목록이 비어 있으면 "예약이 없습니다."와 같은 안내를 표시한다.
  - 각 행에 삭제 버튼을 제공한다. 삭제 버튼 클릭 시, 삭제까지 남은 시간을 표시하는 카운트다운/취소 버튼으로 전환한다.
  - 삭제/삭제취소 클릭 이벤트를 상위 콜백으로 전달해 실제 삭제 여부와 타이머 제어를 상위에서 수행하도록 한다.

- `src/components/StatusBar.jsx`
  - 상위에서 전달받은 메시지의 상태(성공, 실패 등) 따라 하단 상태 바를 표시한다.

- `src/index.css`
  - 비행편 메타 정보, 예약 form, 예약 테이블, 상태 바에 대한 가시성있는 레이아웃을 정의한다.
  - 삭제 카운트다운/삭제취소 상태를 일반 상태와 시각적으로 구분되게 표현한다.

1.5.2 필수 구현 사항

- `setTimeout` 함수를 사용해 **지연 삭제(Undo)**를 구현한다. 삭제 버튼 클릭 시 카운트다운을 시작하고, 만료되면 `DELETE /api/reservations/:id`로 요청을 보낸다. 카운트다운 중 사용자가 버튼 재 클릭으로 삭제를 취소한다면, `clearTimeout`으로 예약된 작업을 무효화하고 UI 상태를 원복한다.

- `setTimeout` 함수를 사용해 **상태 바 자동 숨김**을 구현한다. 상태 바에 메시지를 표시한 뒤 일정 시간이 지나면 자동으로 상태 바를 숨긴다. 새로운 메시지를 표시할 때는 기존 타이머를 `clearTimeout`으로 정리한 후 새 타이머를 설정한다.

1.6 제출물 및 제출 요구사항

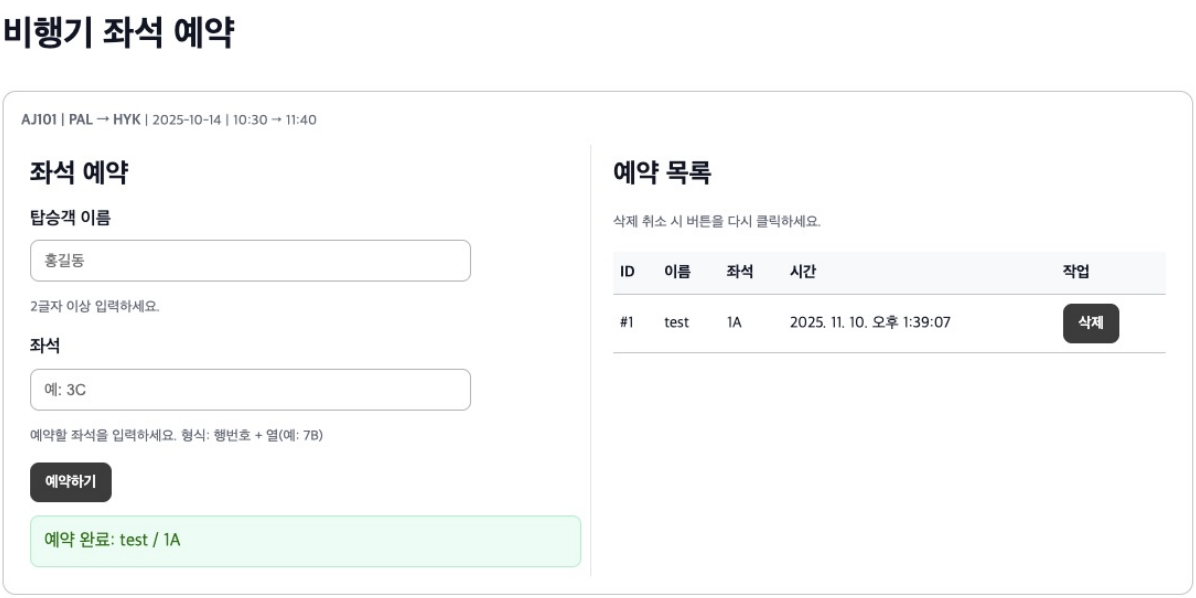
- 구현 코드는 `node_modules` 파일 삭제 후 제출하며, HW#2의 backend 코드도 함께 제출 (미 준수 시 감점 가능)
- 과제 보고서 (`report.pdf`)
  - 상태 관리 및 데이터 흐름 설명: 상위/하위 컴포넌트 간 상태/콜백 전달 방식, props 구조에 대한 설명.
  - 참고한 사이트 목록 (링크 포함).
  - Generative AI 사용 명시: 과제 구현에 있어 AI를 사용했다면 어떤 질문을 했고, 어떤 답변을 받았으며, 코드의 어떤 부분에 사용하였는지 구체적으로 명시.
- 제출물은 코드와 보고서를 HW3_학번_이름.zip 형식으로 압축하여 제출 (미 준수 시 감점 가능)

```
HW3_202301234_홍길동.zip/
├── report.pdf
├── backend/
│   └── HW#2의 backend 코드로 제출 가능
└── frontend/
    └── npm create vite@latest frontend -- --template react 로 생성된 전체 파일
```

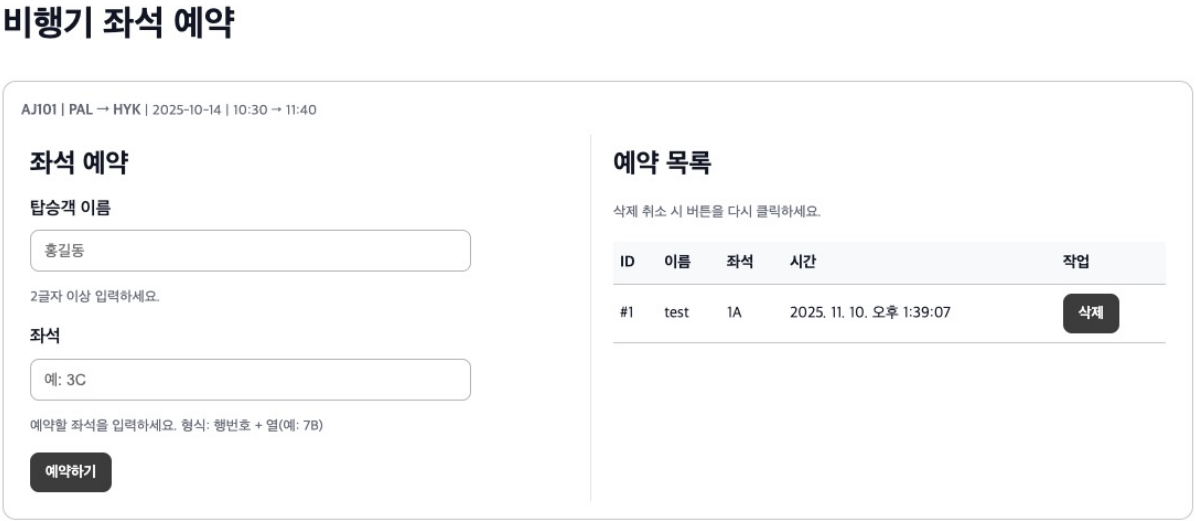
- 제출 기한: 11/18 (화) 23:59 (지각 제출 시 D+1 -40%, D+2 -70%, D+3 -100% 감점)

2 Appendix

과제 코드 실행 예시 결과

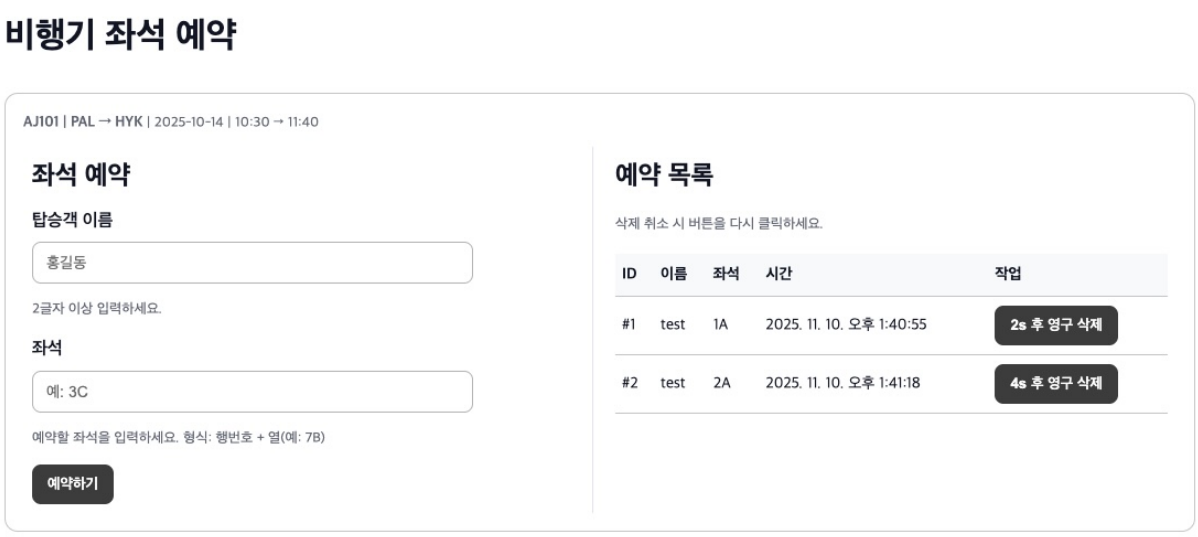


(a) 예약 생성

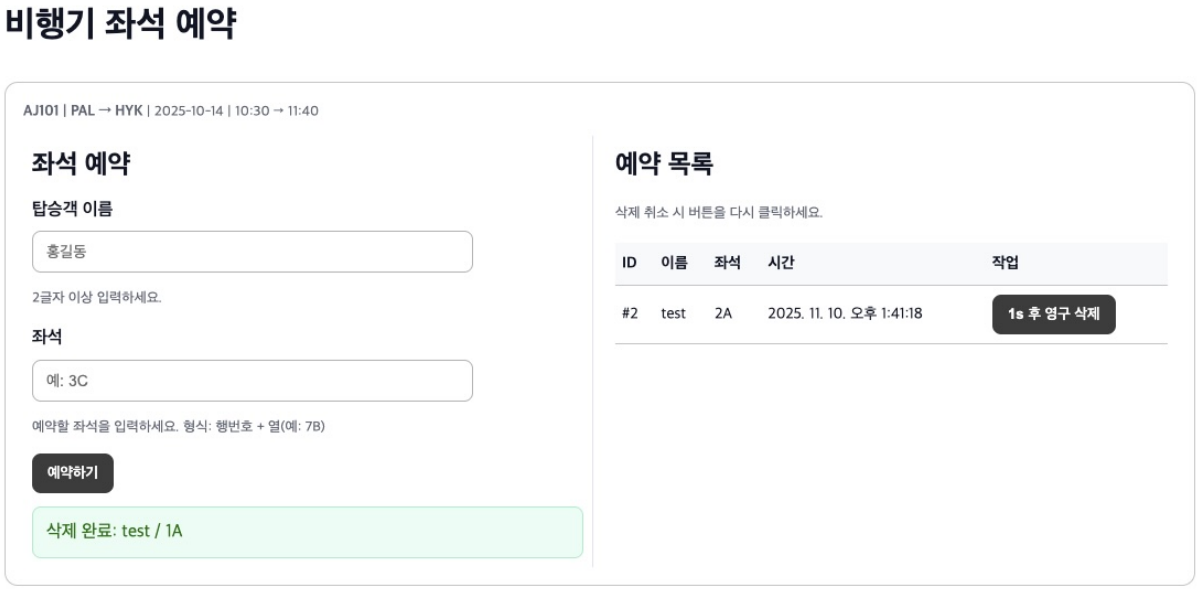


(b) 3초 이후 상태 바 숨김

Figure 1: 예약 생성 및 상태바 처리

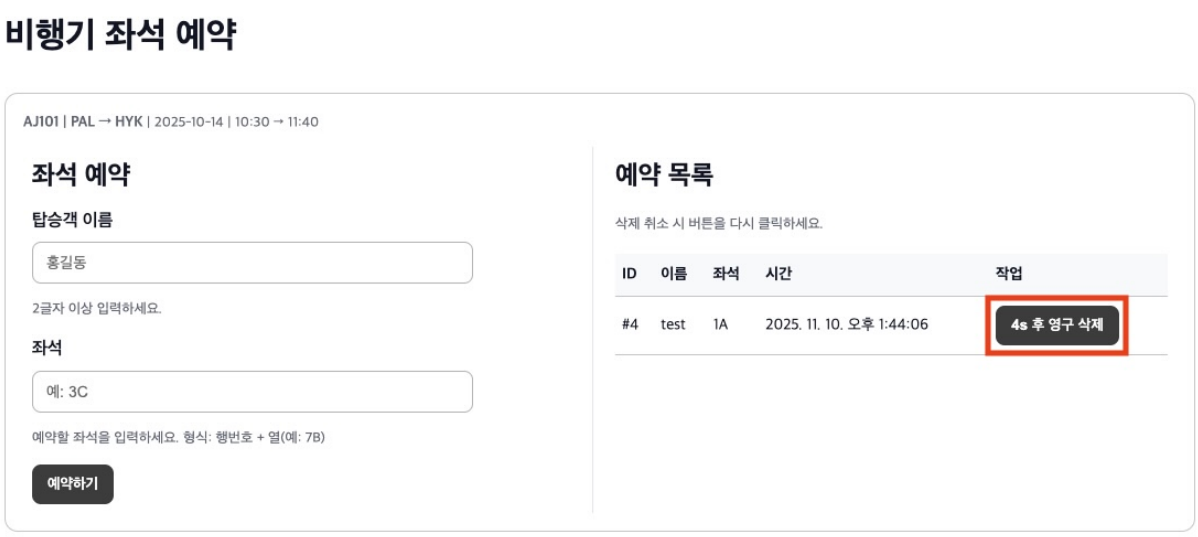


(a) 두 예약을 2초 간격으로 삭제

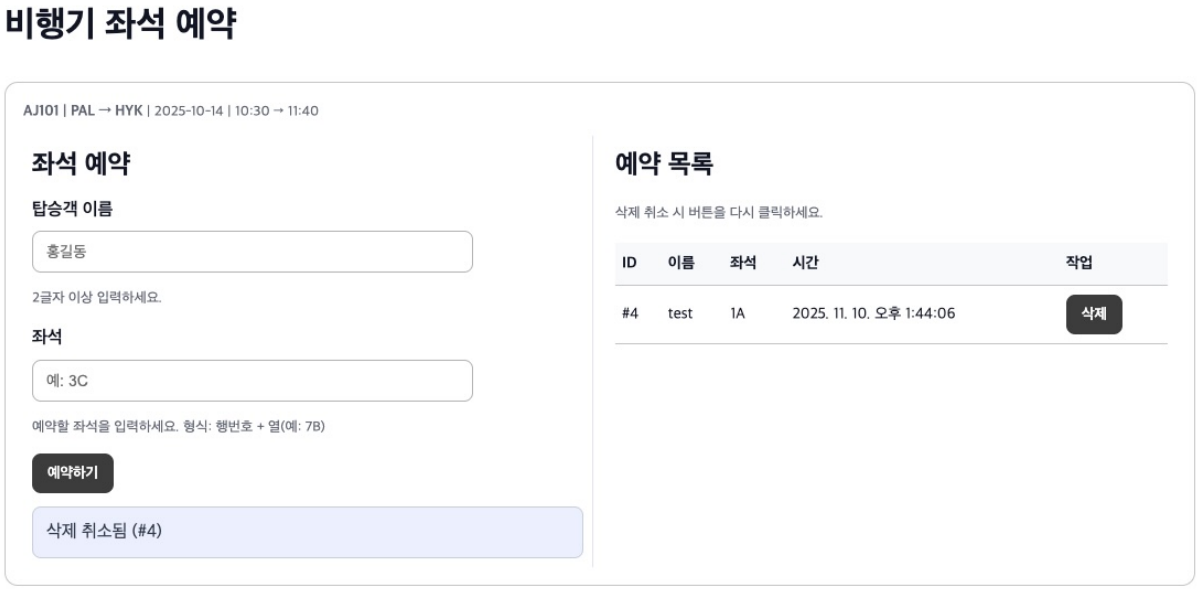


(b) 1개의 예약만 먼저 삭제 처리

Figure 2: 카운트다운 기반 예약 삭제 1



(a) 삭제 취소



(b) 삭제 취소 반영

Figure 3: 카운트다운 기반 예약 삭제 2