React 기반의
웹 애플리케이션 제작 보고서

이름 : 김해민

학번 : 2501110202

담당 교수 : 민경주

과목명 : 웹프로그래밍활용

1. 주제 및 기획 의도

1.1 주제

- 프로젝트명: Echo of Movement (E.O.M) - React Edition

Echo of Movement (E.O.M): 스트릿 댄서들을 위한 올 인원 커뮤니티 플랫폼

1.2 주제 선정 배경

본 프로젝트는 제가 실제 댄서로 활동하며 겪었던 **'소통과 구인의 비효율성'**을 해결하기 위해 기획되었습니다. 현재 댄서 씬은 인스타그램 DM이나 지인 소개 등 비공식적인 경로로 구인·구직이 이루어져, 실력이 있음에도 기회를 얻지 못하거나 정보의 불균형이 심각합니다.

"안무를 함께 짤 팀원", "연습실 쉐어 멤버", "배틀 행사 정보" 등 흩어져 있는 정보, 필요한 정보를 한곳에 모아, 댄서들이 오직 '춤'에만 집중할 수 있는 환경을 제공하고자 이 플랫폼을 기획하게 되었습니다.

2. 사용 기술 및 개발 환경 변화

기존의 정적 웹사이트(HTML/CSS/JS) 방식에서 벗어나, 유지보수와 확장성이 뛰어난 React 웹 애플리케이션으로 재구축하였습니다.

구분	기존 기술	변경된 기술	설명
구조	HTML5 (다중 파일)	React (SPA)	페이지 이동 시 깜빡임 없는 부드러운 사용자 경험 제공
스타일	CSS3 (단일 파일)	CSS Modules	다크 / 라이트 모드 테마 관리 및 컴포넌트별 스타일 분리
로직	Vanilla JS (DOM 직접 제어)	React Hooks	상태(State) 기반의 UI 구현 및 코드 재사용성 확보
라우팅	<a> 태그 이동	React Router	라우팅으로 서버 부하 감소 및 속도 향상

3. 핵심 구현 내용 및 설계 전략

단순히 기존 화면을 리액트로 구현하는 것이 아니라, 리액트의 컴포넌트 재사용성과 효율적인 상태 관리에 중점을 두고 설계하였습니다.

* 레이아웃 디자인도 수정하여 디자인을 향상시키고 싶었으나, 디자인 틀은 기존대로 유지하고 리액트의 구조를 이해하고 설계하는데 집중했습니다.

3.1 디렉토리 구조 및 컴포넌트 분리

기존에는 하나의 HTML 파일에 모든 코드가 들어있어 수정이 어려웠습니다. 이를 해결하기 위해 기능별로 폴더를 나누고 컴포넌트를 분리하였습니다.

- components/layout : 모든 페이지에 공통으로 들어가는 Header, Footer를 분리하여 코드 중복을 제거했습니다.

- components/common : 버튼(Button.jsx)과 같이 반복되는 UI 요소를 별도 컴포넌트로 만들어, 색상이나 크기 등을 props로 받아 재사용할 수 있게 만들었습니다.

- pages : Home, Login 등 각 페이지를 독립된 컴포넌트로 관리하여 라우팅 구조를 명확히 했습니다.

3.2 커스텀 훅을 활용한 로직 분리

컴포넌트 안에 복잡한 로직이 섞여 있으면 코드를 읽기 어렵습니다. 따라서 핵심 기능을 커스텀 훅(Custom Hook)으로 분리하였습니다.

- 타이핑 효과 (useTypewriter): 메인 화면의 "WE DON'T JUST MOVE..." 문구가 타자기처럼 쳐지는 효과를 별도의 훅으로 만들었습니다. 덕분에 어떤 텍스트든 이 훅만 사용하면 타자기 효과를 줄 수 있습니다.

- 테마 관리 (useTheme): 다크 모드와 라이트 모드를 전환하는 로직을 훅으로 만들어, 어느 컴포넌트에서든 쉽게 테마를 변경할 수 있도록 했습니다.

3.3 전역 상태 관리 (Context API)

수업시간에 해본 테마 변경을 참고하여 이번 프로젝트에서는 Context API를 사용하여 '현재 테마 상태'를 앱 전체에 반영했습니다. 덕분에 버튼 하나만 누르면 웹사이트 전체의

색상 테마가 즉시 변경되며, localStorage를 통해 새로고침 후에도 설정이 유지되도록 구현했습니다.

- 구현 및 기술적 난관 극복 : 단순히 상태만 전역으로 관리하는 것이 아니라, 실제 스타일 적용 과정에서 CSS 우선순위문 제와 인라인 스타일 상속 제한 등 예상치 못한 난관에 부딪혔습니다. 특히, 컴포넌트별로 산재된 스타일이 테마 변경 시 즉각적으로 반영되지 않거나, 일부 요소가 기존 스타일을 고수하는 등의 문제가 발생했습니다. 이를 해결하기 위해 ThemeContext에서 테마 상태뿐만 아니라 CSS 변수를 제어하는 로직을 통합하여 관리하도록 설계했습니다. useEffect를 통해 테마 상태가 변경될 때마다 documentElement의 속성을 직접 조작함으로써, 하위 컴포넌트들이 CSS 변수를 통해 자연스럽게 스타일을 상속받도록 구현했습니다.

3.4 동적인 애니메이션

- 스크롤 애니메이션: framer-motion 라이브러리를 도입하여, 사용자가 스크롤을 내릴 때마다 사진과 텍스트가 부드럽게 떠오르는 효과, 옆에서 밀려 나오는 효과 등을 주어 동적인 페이지를 구성하고 시각적인 몰입감을 높였습니다.

3.5 동적인 라우팅과 상태 관리 전략

- react-router-dom을 활용하여 로그인 페이지와 메인 페이지 간의 이동을 매끄럽게 처리하였습니다. 우산 키오스크 프로젝트 수행 경험을 통해, URL 변경이 필요한 페이지 전환과 URL 유지 상태에서의 컴포넌트 재렌더링을 명확히 구분하는 것이 중요하다는 점을 깨달았습니다.

- 구현 방식 : 이에 따라 Home, Login과 같이 페이지 단위의 큰 전환은 라우터를 통해 처리하여 브라우저 히스토리 관리와 직접 접근이 가능하도록 하였고, 동일 페이지 내에서의 섹션 이동이나 모달 등의 작은 변화는 상태 관리(State Management)를 통한 조건부 렌더링으로 처리하여 불필요한 페이지 로드를 방지하였습니다. 추후는 useReducer를 도입해 더 체계적인 관리를 하고자 합니다.

4. 결론 및 향후 계획

이번 과제를 통해 정적인 웹사이트를 동적인 리액트 애플리케이션으로 전환했습니다. 이 과정에서 왜 리액트를 사용하는가?"에 대한 답을 얻을 수 있었습니다. 컴포넌트 단위로 개발하니 유지보수가 훨씬 쉬워졌고, 상태 관리를 통해 페이지별로 전부 새로 생성할

필요 없이 복잡한 UI도 효율적으로 제어할 수 있었습니다.

현재는 프론트엔드 기능 위주로 구현되어 있지만, 향후에는 백엔드 서버와 연동하여 실제 댄서들이 회원가입을 하고 게시글을 올릴 수 있는 실전형 커뮤니티로 발전시킬 수 있다면 좋겠습니다.

5. 실제 구현 화면

: 이 부분은 링크로 대체합니다. 레포짓토리의 ReadME에서 확인하시거나, 아래 LiveSite로 접속하시면 실제 페이지를 체험 해보실 수 있습니다.

Github Repository: https://github.com/hazyala/E.O.M_Web_React.git

Live Site : https://hazyala.github.io/E.O.M_Web_React