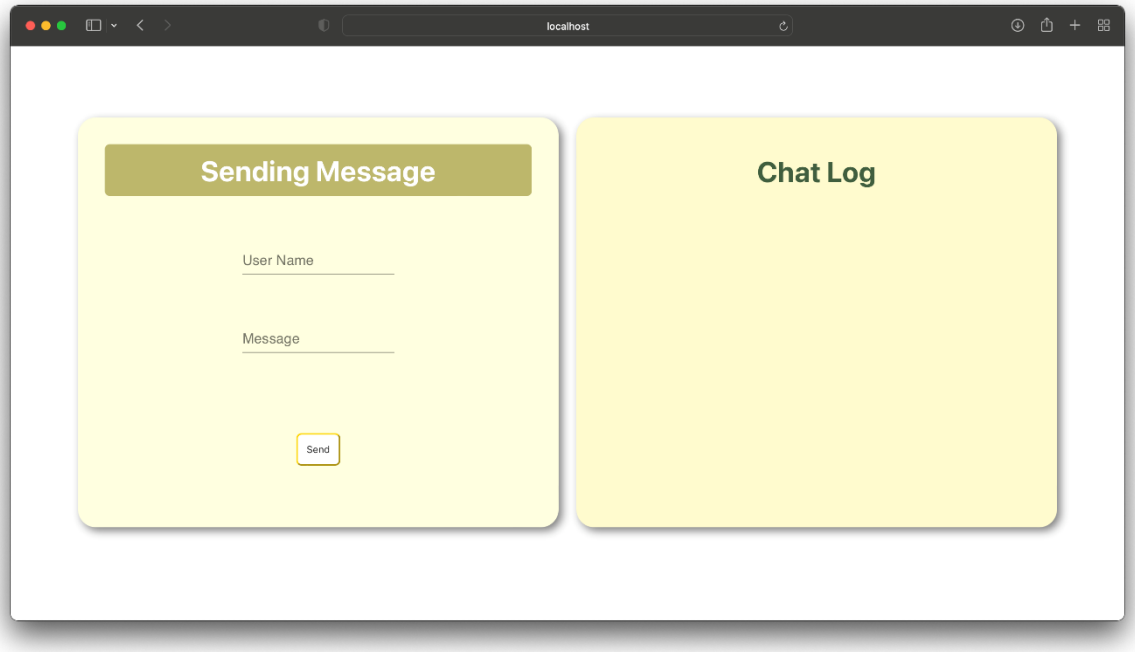
8주차 과제

20161663 허재성

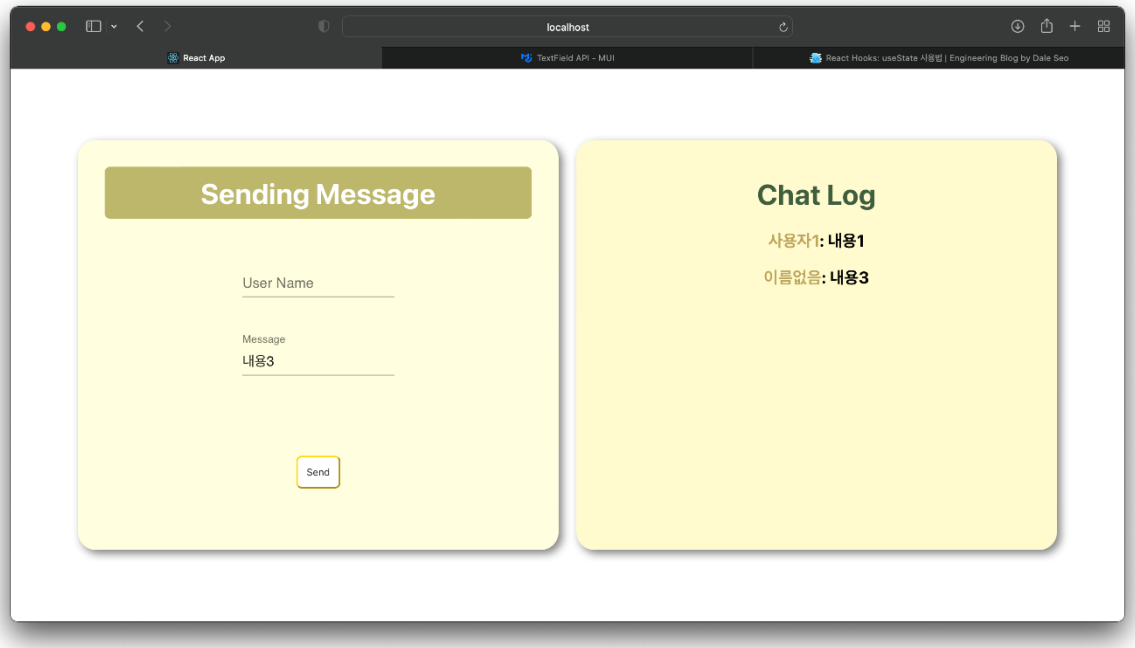
8주차 실습에서는 채팅 앱을 개발하기 위해 총 4개의 컴포넌트가 존재한다. 구현하려는 채팅 앱의 다음과 같은 화면을 가진다.



먼저 화면 좌측의 제목 Sending Message와 입력창 User Name, Message, 그리고 send 버튼을 포함한 부분을 나타내는 컴포넌트는 MessengerTemplate 컴포넌트이다. MessengerTemplate 컴포넌트는 Title 태그와 UserName 태그, MessageContents 태그, Button 태그를 Form 태그로 묶어서 return한다. Title 태그는 Sending Message라는 제목이 적힌 태그이다. UserName 태그는 채팅의 사용자 이름을 입력하는 태그이고, MessageContents는 채팅 메시지를 입력하는 태그로 두 태그는 Material UI의 TextField 컴포넌트를 감싸서 구현되었다. Button 태그는 메시지를 전송하는 버튼을 구현하였다.

Title, Form, UserName, MessageContents, Button 태그는 styled-components를 이용해 CSS-in-JSX로 스타일이 적용되었다.

화면 우측의 Chat Log 부분을 나타내는 컴포넌트는 ChatLogTemplate 컴포넌트이다. ChatContainer라는 태그를 반환한다. ChatContainer 태그는 Chat Log라는 제목과 chats 배열에 저장된 채팅 내용을 리스트 형태로 반환하는 JSX 함수를 감싼다. chats 배열의 내용을 형식에 맞게 반환하기 위해 ChatLogItem 컴포넌트가 존재하는데 이에 대해선 후술하겠다. ChatContainer 태그 또한 styled-components를 이용해 CSS-in-JSX로 스타일이 적용되었다.



우측의 Chat Log 화면에 채팅 내역이 존재하는 것을 알수 있다. 이 때 채팅 리스트(사용자1:내용1(줄바꿈)이름없음:내용3)에서 사용자1:내용1이 하나의 채팅이다. 이러한 하나의 채팅을 나타내는 컴포넌트가 ChatLogItem 컴포넌트이다. ChatLogItem 컴포넌트에서는 username 문자열과 message 문자열을 field로 가지는 chat 객체를 받아서 UsernameContainer 태그로 감싸고 message 부분은 ChatContainer 태그로 감싸서 반환한다. ChatContainer로 감싸진 message 부분은 검은색 텍스트로, UsernameContainer로 감싸진 username은 #bda55d(노란색) 텍스트로 화면에 나타난다.

마지막으로 ChatLogTemplate와 MessengerTemplate를 감싸는 컴포넌트로 ChatTemplate가 존재한다. 전체 화면을 관리하는 컴포넌트로 MessengerTemplate와 ChatLogTemplate를 Container 태그로 묶어서 return한다. 또한 ChatTemplate 컴포넌트 내에는 getChatLog 함수를 정의하여 chats 배열에 새로운 username과 message를 갖는 chat 객체를 추가할 수 있게 하고, getChatLog를 MessengerTemplate에 props로 전달하여 MessengerTemplate에서 getChatLog를 호출하여 새로운 채팅을 추가할 수 있도록 하였

다.

이번 실습의 핵심은 화면 왼쪽에 username과 message를 입력했을 때, 입력한 내용이 chat 객체인 MsgState에 잘 저장되고 send 버튼을 눌렀을 때 getChatLog 함수를 통해 chat 객체가 chats 배열로 전달되어 ChatLogTemplate에 의해 우측 화면에 리스트 형태로 나타나게 하는 것이다. MessemgerTemplate에서 좌측 화면의 username과 message를 입력받을 수 있게 UserName 태그 안에 TextField 태그를 추가하였다. 이 때 TextField로 생긴 입력창에 키보드로 입력했을 때 문자열이 바뀌는 입력을 위해 onChange 속성으로 onUsernameChange call back 함수를 추가한다. 마찬가지로 MessageContents 태그 안에도 TextField를 추가 후, onChange 속성으로 onMessageChange call back 함수를 추가한다. Button 태그에는 버튼을 눌렀을 때 chat 객체가 chats 배열에 추가되는 것을 구현하기 위해 onClick 속성으로 onMsgSubmit call back 함수를 추가한다. Call back 함수 onUsernameChange는 사용자가 username 입력창에 텍스트를 입력하면 입력된 내용을 MsgState의 username에 저장한다. 비슷하게 onMessageChange는 사용자가 message 입력창에 텍스트를 입력하면 입력된 내용을 MsgState의 message에 저장한다. 이로 인해 username과 message 창에 키보드로 입력하면 chat 객체의 username과 message에 가각 입력한 내용이 저장되게 된다. 이 상태에서 다시 입력하여 내용을 수정할 수 있다. 만약 사용자가 화면의 send 버튼을 누르면 call back 함수 onMsgSubmit이 호출된다. onMsgSubmit에서는 먼저 MsgState의 message가 빈 문자열인지 확인한다. 조건문으로 확인하여 만약 빈 문자열이면 alert 함수로 "메시지를 입력하세요." 경고창을 띄운 후 return하여 아무 일도 일어나지 않게 한다. 이 경우에는 chats 배열에 입력한 chat 객체가 추가되지 않는다. 만약 message가 빈 문자열이 아니면 다음으로 MsgState의 username을 확인한다. Username이 빈 문자열이면 "이름없음"으로 정해주고 그렇지 않으면 사용자가 입력한 그 텍스트를 username으로 정한다. 결국 사용자가 message 내용을 입력하지 않았다면 경고창을 띄워 message를 입력하게 강제하고, message는 입력했다면 username의 입력 유무에 상관 없이(입력하지 않았다면 "이름없음"으로 고정) getChatLog 함수를 호출해 chats 배열에 추가해주고 이로 인해 우측 화면에서 채팅 내용을 확인할 수 있게 된다. Chats 배열에 추가 후 다음 입력을 위해 MsgState의 username과 message를 모두 빈 문자열로 reset하면 onMsgSubmit의 역할은 끝난다.