**계정 만드는 방법에 대한 챕터**

1. 참고 <https://firebase.google.com/docs/reference/js/firebase.auth.Auth?authuser=2#setpersistence>
2. firebase.auth 안에 수많은 authProvider가 있음. 원하는 걸 골라서 사용.
3. 여기선 EmailAuthProvider를 기준으로 설명
4. 중요한 것 메소드 두개 : createUserWithEmailAndPassword, signInWithEmailAndPassword
5. Auth.js 안에 다음처럼 코드 작성

import { authService } from 'myFirebase';

import React, { useState } from 'react';

const Auth = () =>{

…

**const [newAccount, setNewAccount] = useState(true);**

…

**const onSubmit = async(event) => {**

**//이벤트 기본 행위를 막기위한 함수 삽입**

**event.preventDefault();**

**try {**

**let data;**

**if(newAccount) {**

**//create account**

**data = await authService.createUserWithEmailAndPassword(email, password);**

**} else {**

**// log in**

**data = await authService.signInWithEmailAndPassword(email, password);**

**}**

**console.log(data);**

**} catch (error) {**

**console.log(error);**

**}**

**}**

return (<div>

**<form onSubmit={onSubmit}>**

…

**<input type="submit" value={newAccount ? "Create Account" : "Log In"} />**

</form>

1. 설명
   1. Log In 상태와 Create Account 상태를 분리하기 위해 useState로 newAccount를 생성
   2. form 안의 submit으로 돼있는 input을 newAccount가 참이면 Create Account를 아니면 Log In을 보이도록 수정해줌.
   3. form 에 onSubmit에 onSubmit 함수를 넣어 submit되면 onSubmit 함수를 실행토록 함
   4. onSubmit 함수 안에는 기본 행위를 막는 메소드를 넣고, newAccount가 참이면 계정을 등록하고, 아니면 로그인하도록 수정함
   5. 이때 onSubmit 함수의 인자는 async로 감싸고, (await 를 쓰면 async로 감싸야하는 듯)

authService.createUserWithEmailAndPassword(email, password)

는 앞에 await를 붙여주자.(Promise 를 주므로?)

* 1. 이때, createUserWithEmailAndPassword와 signInWithEmailAndPassword는 에러를 던질 수 있기에 try문으로 감싸준다.
  2. 그리고 위 두 메소드가 무슨 값을 던지는지도 확인하기 위해여 data를 받아 두 함수가 종료후 console.log(data)를 출력하도록 하자.

1. 여기까지 했을 때, 계정은 실제로 만들어진다. 파이어베이스 콘솔로 가면 확인할 수 있다.
2. 하지만 아직 로그인 한 후의 화면을 만들지 않아서 바뀐 게 없는 것처럼 보이는 거다. 이건 다음 챕터에서
3. setPersistance를 통해 persistance 설정도 해줄 수 있다.

local : 브라우저를 닫더라도 사용자 정보가 저장됨

session : 현재 탭이 켜져있는 동안 사용자 정보가 저장됨

none : 사용자정보를 저장하지 않음. 브라우저를 새로고침하거나, 다른 페이지에 가면 로그아웃