**로그인 되면 로그인 된 후 홈 화면으로 보내주는 작업을 해주는 챕터**

1. 기존 코드의 문제점 : 로그인 했는데도 홈화면으로 안감.
   1. 이유 : 다음에서 useState 안에 authService.currentUser가 있어서 이게 isLoggedIn으로 들어가서 로그인 됐는지 판별하는데, 이 useState 코드가 실행되는데 시간이 너무 짧아서 firebase가 실행되기 전에 currentUser를 참조하여 당연히 false가 나와서 넘어가지 않은 것.

const [isLoggedIn, setIsLoggedIn] = useState(false);

  return (<>

    <AppRouter isLoggedIn={isLoggedIn}></AppRouter>

    <footer>&copy; { new Date().getFullYear()} Nwitter</footer>

    </>

  );

1. 따라서 firebase를 기다려주는 코드를 짜자.
   1. isLoggedIn과 연결된 useState 안을 false로 바꾸자.
   2. 컴포넌트가 실행되면 바로 실행되는 useEffect를 선언하여 안에 .onAuthStateChanged 메소드를 추가하자.

<https://firebase.google.com/docs/reference/js/firebase.auth.Auth?authuser=2#onauthstatechanged>

* 1. 자세한 정보는 위 링크 안에 있다.
  2. 간단하게, 사용자의 로그인 상태의 변화를 관찰하는 관찰자이다. 로그인 했을 때, 로그아웃했을 때 등등. 로그인 돼있을 때 true를 user에 넣는 거 같음
  3. 따라서 onAuthStateChanged 안에 함수를 선언하여 user가 true일 때 isLoggedIn을 true로 하고, false일 때 false로 하자.

const [isLoggedIn, setIsLoggedIn] = useState(false);

  useEffect(() => {

    authService.onAuthStateChanged((user) => {

      if(user) {

        setIsLoggedIn(true);

      } else {

        setIsLoggedIn(false);

      }

    });

  }, []);

  return (<>

    <AppRouter isLoggedIn={isLoggedIn}></AppRouter>

    <footer>&copy; { new Date().getFullYear()} Nwitter</footer>

    </>

  );

1. 2번까지하면 로그인됐을 때 홈화면으로 이동한다.

**홈화면 이동 중일 때 처리**

1. 로그인이 됐거나, 혹은 되지 않아 IsLoggedIn이 설정되고, 어디로든 이동할 때 초기화 화면을 보여주자.
   1. 아래처럼 처음엔 false로 선언하였다가, 로그인 상태 변화가 한번이라도 일어나면 init을 true로 바꾼다.
   2. 그리고 AppRouter 렌더링 부분에서 init이 false면 initalizing을 보이도록 하고 true일 때만 렌더링 하도록 한다.

 const [init, setInit] = useState(false);

  const [isLoggedIn, setIsLoggedIn] = useState(false);

  useEffect(() => {

    authService.onAuthStateChanged((user) => {

      if(user) {

        setIsLoggedIn(true);

      } else {

        setIsLoggedIn(false);

      }

      setInit(true);

    });

  }, []);

  return (<>

    {init ? <AppRouter isLoggedIn={isLoggedIn}></AppRouter> : "Initializing"}

    <footer>&copy; { new Date().getFullYear()} Nwitter</footer>

    </>

  );

**로그인 에러처리**

1. 로그인 시 에러 발생(아이디 중복, 비밀번호 자리수 부족 등)시, 안내 메시지를 옆으로 보이도록 설정
2. auth.js 파일에서 error를 useState로 만들어주자.

const [error, setError] = useState("");

1. auth.js 파일에서 onSubmit 시 에러 처리부분을 수정해주자

catch (error) {

            setError(error.message);

        }

1. 그리고 적절한 곳에 렌더링 해주자

{error}

**계정 만들기랑 로그인 버튼 바꾸기**

1. 로그인 하고 싶으면 로그인 버튼 누르면 로그인하도록, 반대로 계정만들고 싶으면 계정 만들 수 잇도록 하기.
2. 다음 toggle 함수를 만들어서, 버튼을 누르면 newAccount 변수가 바뀌도록 함.

 const toggleAccount = () => setNewAccount(prev => !prev);

1. 그리고 적절한 곳에 다음을 렌더링

<span onClick={toggleAccount}>{newAccount ? "Sign In" : "CreateAccount"}</span>

1. 이때 이미 버튼으로 있는 newAccount 와는 반대로 설정해야한다.
2. 전체코드는 다음과 같음.

    …

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

…

    const onSubmit = async(event) => {

        //이벤트 기본 행위를 막기위한 함수 삽입

        event.preventDefault();

        try {

            let data;

            if(newAccount) {

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

            } else {

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

            }

            console.log(data);

        } catch (error) {

            setError(error.message);

        }

    }

    const toggleAccount = () => setNewAccount(prev => !prev);

    return (<div>

        <form onSubmit={onSubmit}>

…

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

        </form>

        <span onClick={toggleAccount}>{newAccount ? "Sign In" : "CreateAccount"}</span>

       …

    </div>

    );

};

export default Auth;