K-Digital Training 웹 풀스택 과정

JWT 인증

OAuth

OAuth란?

OAuth(Open Authorization)

- 인터넷 사용자들이 **비밀번호를 제공하지 않고 다른 웹사이트 상의 자신들의 정보에 대해 웹사이트나 어플리케이션의 접근 권한을 부여할 수 있는 공통적인 수단으로서** 사용되는, 접근 위임을 위한 개방형 표준이다(위키백과)

- 즉, 서드파티 어플리케이션이 사용자의 계정에 접근할 수 있는 권한을 부여하기 위한 프로토콜

OAuth

OAuth가 사용되기 전에는 인증방식의 표준이 없었기 때문에 기존의 기본인증인 아이디와 비밀번호를 사용하였는데, 이는 보안상 취약한 구조일 가능성이 매우 높았고 기본 인증이 아닐 경우는 각 애플리케이션들이 각자의 개발한 회사의 방법대로 사용자를 확인하였다.

OAuth 2.0

- OAuth 2.0은 보다 간편하고 확장성이 있는 버전으로, 기본적으로 인증과 권한을 분리하여 다루는 것을 특징
- **Access Token**을 통해 권한을 부여하고, 사용자의 실제 비밀 정보를 공유하지 않습니다.

OAuth 2.0

JWT Bearer Token 인증

- OAuth 2.0에서는 Bearer Token 인증 방식을 지원
- JWT를 Bearer Token으로 사용할 수 있음
- 클라이언트가 발급받은 JWT를 "Authorization" 헤더에 담아 요청을 보내면, 서버는 해당 JWT의 유효성을 검사하여 권한을 부여

```
Authorization: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VySWQiOjE
```

* Bearer? 웹 서비스에서 사용자 인증에 사용되는 인증 방식 중 하나

JWT

JWT란?

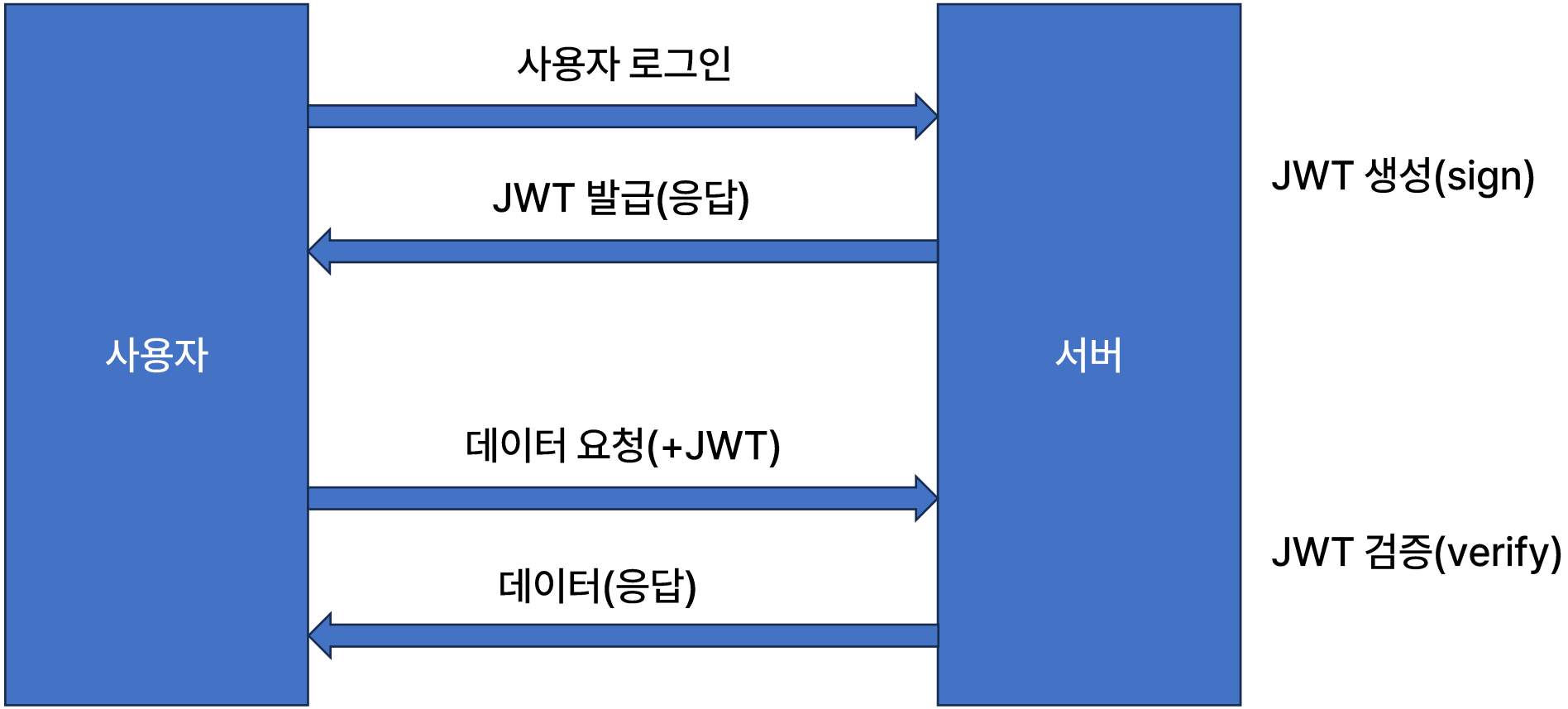
JWT(JSON Web Token)

- 웹 애플리케이션에서 정보를 안전하게 전송하기 위한 간단한 방법
- 인증된 사용자를 식별하거나 데이터에 서명을 하여 변조를 방지하는 데 사용

JWT 구성

- **Header**: 토큰의 타입과 해시 알고리즘 정보가 포함
- **Payload**: 실제 정보 데이터가 포함되며, 클레임(claim)이라 불리는 키-값 쌍으로 이루어져 있음
- **Signature**: 토큰의 무결성을 검증하기 위한 서명 부분으로, Header와 Payload의 조합에 비밀 키를 사용해 생성

JWT 작동방식


사용자
서버
사용자 로그인
JWT 발급(응답)
데이터 요청(+JWT)
데이터(응답)
JWT 생성(sign)
JWT 검증(verify)

JWT 주요함수

- sign: 서버측에서 JWT를 생성할 때 사용

```
jwt.sign(payload, secretOrPrivateKey, [options, callback])
```

- verify: 클라이언트나 서버에서 받은 JWT의 유효성을 검증할 때 사용

```
jwt.verify(token, secretOrPublicKey, [options, callback])
```

JWT 사용하기

```
npm install jsonwebtoken
```

```
const jwt = require('jsonwebtoken');
```