JWT API 구현

황다인

20211

CONTENTS

웹사이트 로그인을 유지시키는 방법

Cookie

Session

Token

웹사이트 로그인을 유지시키는 방법

Cookie

공개 가능한 정보를 사용자의 브라우저에 저장

장점:

- 간단한 구현
- 서버 자원 절약

단점:

- 보안에 취약
- 용량 제한

웹사이트 로그인을 유지시키는 방법

Session

비밀번호 등 클라이언트의 민감한 인증 정보를
브라우저가 아닌 서버 측에 저장하고 관리

장점: 높은 보안성

단점: 서버의 부하가 심하다

웹사이트 로그인을 유지시키는 방법

Token

토큰을 발급하여 사용자의 신원을 증명

장점: 플랫폼 독립성
단점: 서버 자원 소모

JWT란 무엇인가

Token

→

JWT
(Json Web Token)

JWT란 무엇인가

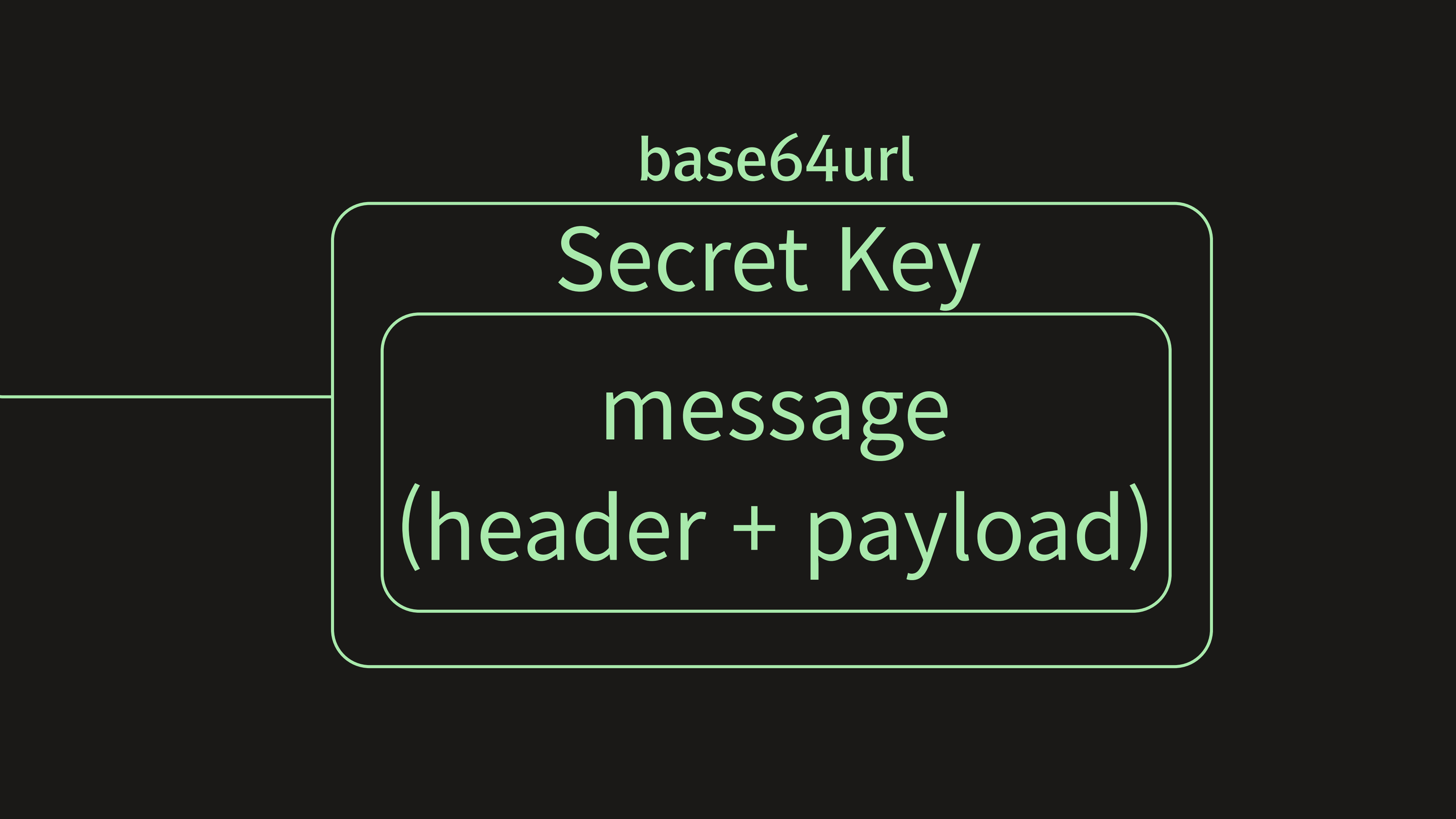
JWT의 구조

header.payload.signature

base64url

```
{
    "alg": 알고리즘 종류
     "typ": "JWT"
}
```

```
{
    "uid": "admin"
     "rank": "admin"
     "exp": 600
}
```


base64url
Secret Key
message
(header + payload)

JWT는 어떻게 토큰을 검증하는가

signature ≠ 암호화

↓

그렇다면 어떻게??

JWT는 어떻게 토큰을 검증하는가

header + payload + Secret key
→ signature

old_signature == new_signatue

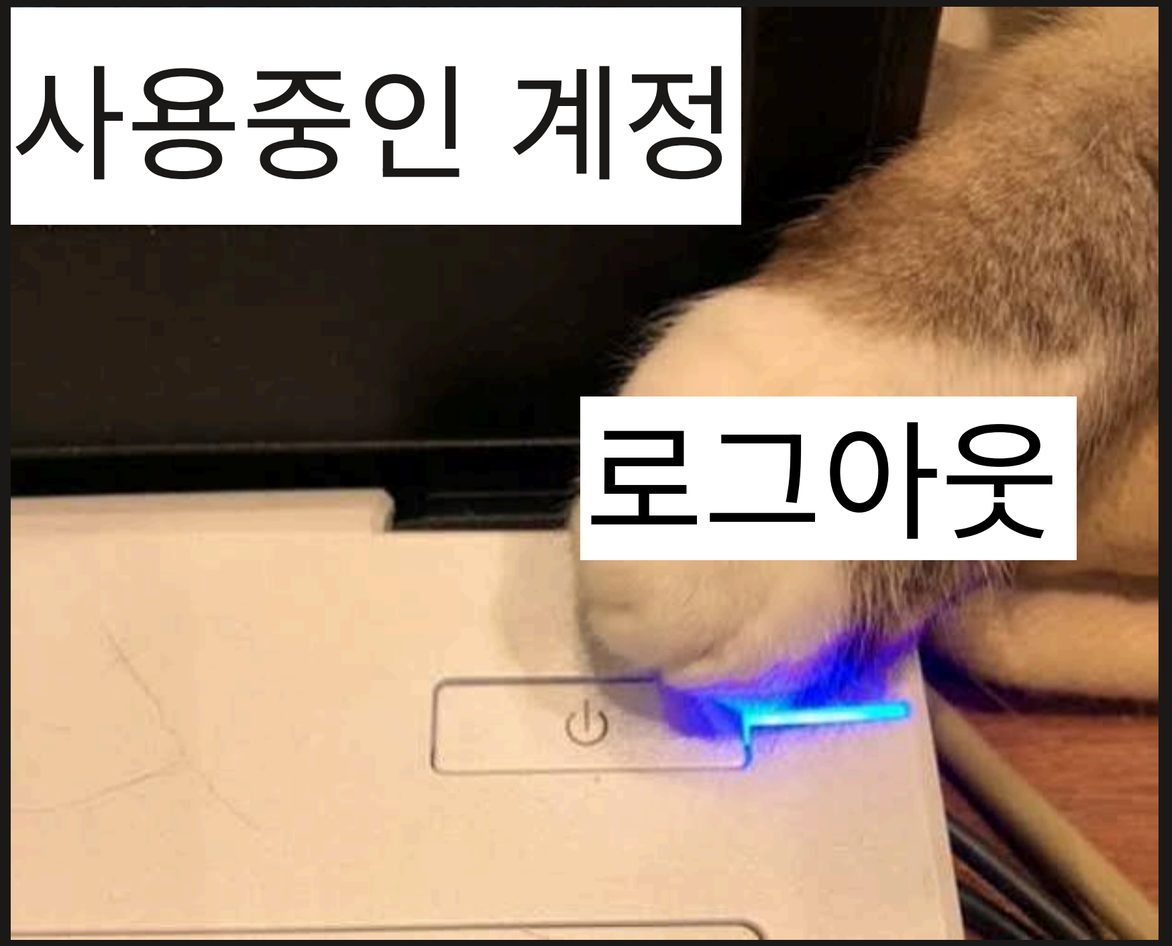
JWT를 저장하는 법

HttpOnly Cookie

1. JS가 읽지 못함
2. CSRF 위험 감소
3. https에서만 전송

이래도 유출이 된다면..?

토큰 만료 시간↓ →



Refresh Token

PROJECT

1. JWT 발급
2. 리프레쉬 토큰
3. API 사용해보기
4. 시연

01
JWT 발급

```
def mkjwt(input: Payload | Token):
    if isinstance(input, Payload):
        bytes = json.dumps(header).encode('UTF-8')
        result = base64.urlsafe_b64encode(bytes)
        result_str1 = result.decode('ascii')

        bytes = input.model_dump_json().encode('UTF-8')
        result = base64.urlsafe_b64encode(bytes)
        result_str2 = result.decode('ascii')
    else:
        result_str1 = input.header
        result_str2 = input.payload
message = f'{result_str1}.{result_str2}'.encode('UTF-8')
signature = hmac.new(secret_key, message, hashlib.sha256).digest()
signature_encoded = base64.urlsafe_b64encode(signature)
.decode('ascii').strip("=")
return result_str1, result_str2, signature_encoded
```

01
JWT 발급

```
def mkjwt(input: Payload | Token):
    if isinstan
        bytes =
        result =
        result_

        bytes =
        result =
        result_
    else:
        result_
        result_
message =
signature =                                                    56).digest()
signature_e                                                    e)
.decode('ascii').strip('=')
return result_str1, result_str2, signature_encoded
```

```
class Payload(BaseModel):
    username: str
    rank: str
    exp: int
class Token(BaseModel):
    header: str
    payload: str
    signature: str
```

01
JWT 발급

```
def mkjwt(input: Payload | Token):
    if isinstance(input, Payload):
        bytes = json.dumps(header).encode('UTF-8')
        result = base64.urlsafe_b64encode(bytes)
        result_str1 = result.decode('ascii')

        bytes = input.model_dump_json().encode('UTF-8')
        result = base64.urlsafe_b64encode(bytes)
        result_str2 = result.decode('ascii')
    else:
        result_str1 = input.header
        result_str2 = input.payload
message = f'{result_str1}.{result_str2}'.encode('UTF-8')
signature = hmac.new(secret_key, message, hashlib.sha256).digest()
signature_encoded = base64.urlsafe_b64encode(signature)
.decode('ascii').strip("=")
return result_str1, result_str2, signature_encoded
```

02 REFRESH 토큰

```
def insert_refresh_token(uid, refresh_token):
    connection = pymysql.connect(**DB_CONFIG)
    with connection.cursor() as cursor:
        sql_delete = '''DELETE FROM refresh_tokens
                        WHERE uid = %s'''
        cursor.execute(sql_delete, (uid,))
        sql_insert = 'INSERT INTO refresh_tokens (uid, refresh_token)
VALUES (%s, %s)'
        cursor.execute(sql_insert, (uid, refresh_token))

    connection.commit()
    connection.close()
    return refresh_token
```

02 REFRESH 토큰

secrets.token_hex(10))

↓

```
def insert_refresh_token(uid, refresh_token):
    connection = pymysql.connect(**DB_CONFIG)
    with connection.cursor() as cursor:
        sql_delete = '''DELETE FROM refresh_tokens
                        WHERE uid = %s'''
        cursor.execute(sql_delete, (uid,))
          sql_insert  =  'INSERT  INTO  refresh_tokens  (uid,  refresh_token)
VALUES (%s, %s)'
        cursor.execute(sql_insert, (uid, refresh_token))

    connection.commit()
    connection.close()
    return refresh_token
```

02 REFRESH 토큰

```
def check_refresh_token(uid, refresh_token):
    connection = pymysql.connect(**DB_CONFIG)
    with connection.cursor() as cursor:
 sql  =  '''SELECT  refresh_token  FROM  refresh_tokens
WHERE uid = %s'''
        cursor.execute(sql, (uid,))
        result = cursor.fetchone()
        if result:
            connection.close()
            return result['refresh_token'] == refresh_token
```

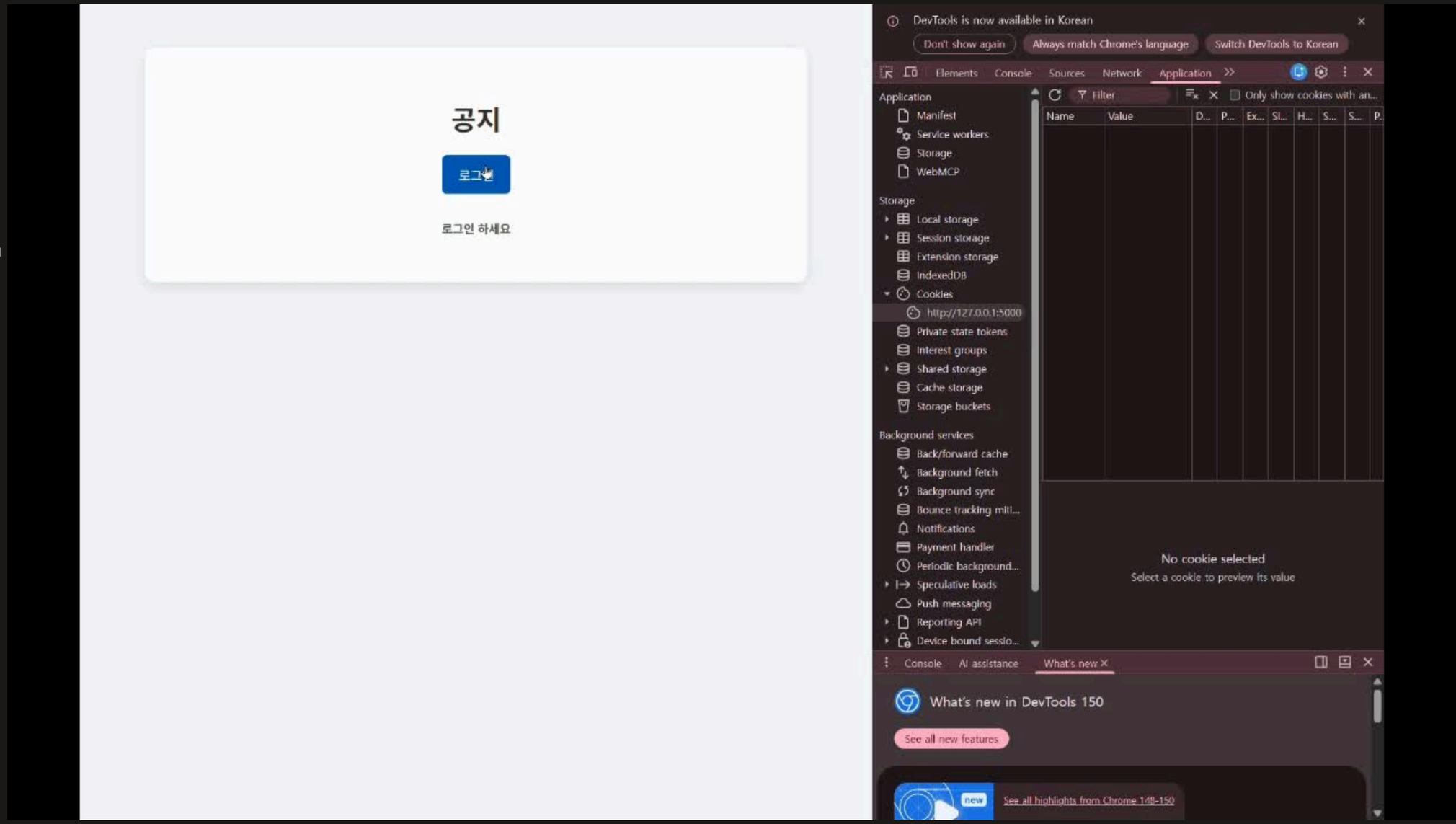
03
API
사용하기

```
@app.post("/token")
def token(response: Response, user: Users):
    result = db.get_users(user.uid)
    if result['pwd'] == user.pwd:
                        cookies.jwt_cookie(response,   result['uid'],
result['class'])
        cookies.refresh_cookie(response, result['uid'])
        return '성공'
    else:
return
Response(status_code=status.HTTP_401_UNAUTHORIZED)
```

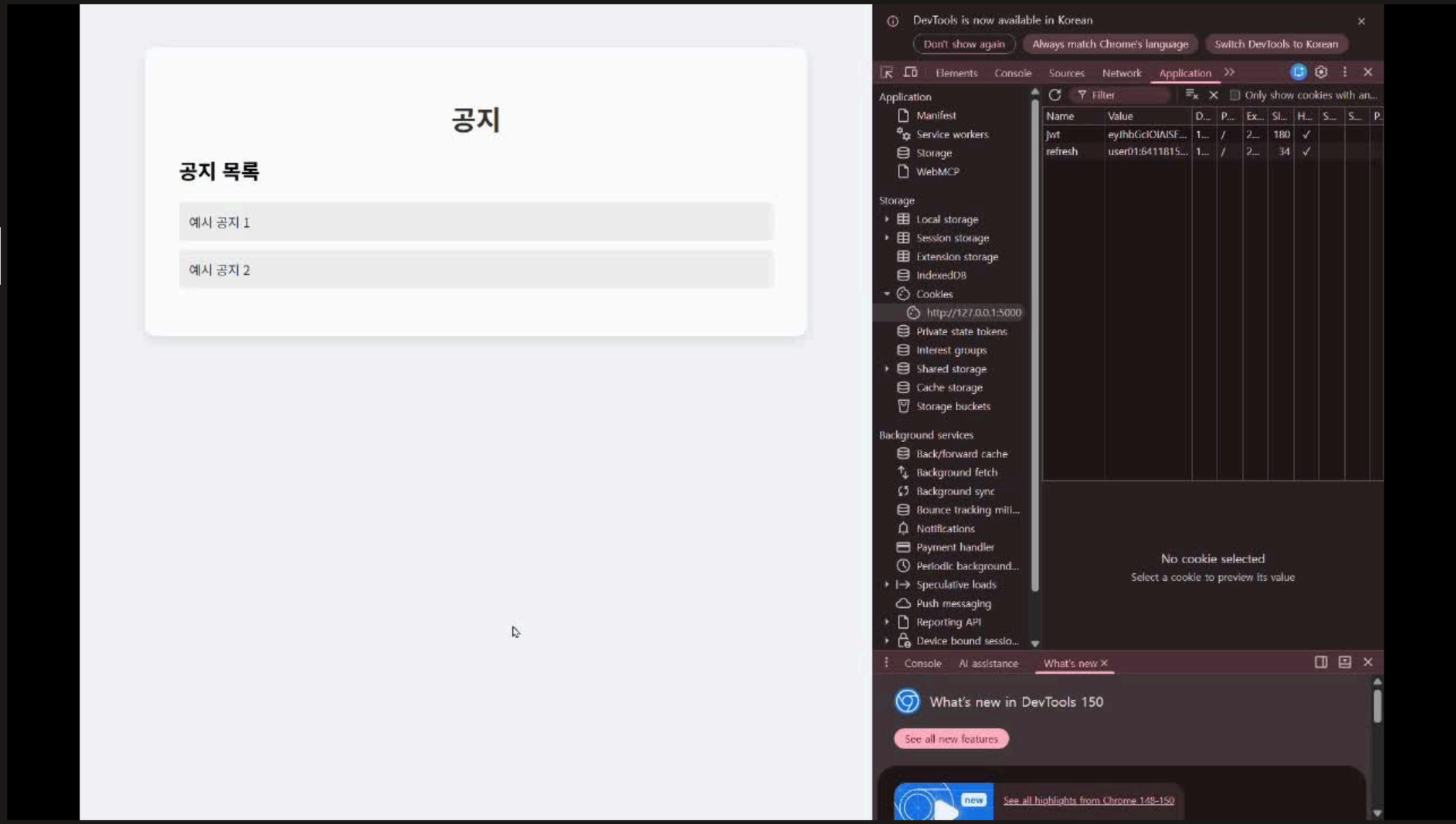
03
API 사용하기

```python
@app.post("/refresh")
def refresh(response: Response, refresht: RefreshToken):
    uid, refre = refresht.refresh_token.split(':')
    if db.check_refresh_token(uid, refre) == True:
        cookies.jwt_cookie(response, uid)
        return "성공"
    return
    Response(status_code=status.HTTP_401_UNAUTHORIZED)
```

04
시연
토큰 발급


공지
로그인
로그인 하세요
DevTools is now available in Korean
Don't show again
Always match Chrome's language
Switch DevTools to Korean
Elements
Console
Sources
Network
Application
Application
Manifest
Service workers
Storage
WebMCP
Storage
Local storage
Session storage
Extension storage
IndexedDB
Cookies
http://127.0.0.1:5000
Private state tokens
Interest groups
Shared storage
Cache storage
Storage buckets
Background services
Back/forward cache
Background fetch
Background sync
Bounce tracking miti...
Notifications
Payment handler
Periodic background...
Speculative loads
Push messaging
Reporting API
Device bound sessio...
Filter
Only show cookies with an...
Name
Value
No cookie selected
Select a cookie to preview its value
Console
AI assistance
What's new
What's new in DevTools 150
See all new features
new
See all highlights from Chrome 149–150

04
시연
REFRESH



느낀점

사용량이 많은 사이트가 아니라면 세션을 사용할것 같다.
사용가능한 API를 만들기가 힘들었다.
AWS를 공부해서 다른 컴퓨터에서도
사용가능한 API를 만들고 싶다.

Q&A

들어주셔서
감사합니다