이 코드는 time 모듈, HTTPException 클래스 등을 FastAPI에서 불러온다. 또한 JWT를 인코딩, 디코딩하는 jose 라이브러리와 Settings 클래스도 불러온다. jose 라이브러리가 설치되어 있지 않다면 다음 명령을 사용해 설치하자.

```
pip install python-jose[cryptography] python-multipart
```

SECRET_KEY 변수를 추출할 수 있도록 Settings 클래스의 인스턴스를 만들고 토큰 생성용 함수를 정의한다.

```
settings = Settings()

def create_access_token(user: str):
    payload = {
        "user": user,
        "expires": time.time() + 3600
    }

    token = jwt.encode(payload, settings.SECRET_KEY, algorithm="HS256")
    return token
```

토큰 생성 함수는 문자열 하나를 받아서 payload 딕셔너리에 전달한다. payload 딕셔너리는 사용자명과 만료 시간을 포함하며 JWT가 디코딩될 때 반환된다. expires값(만료 시간)은 생성 시점에서 한 시간 후로 설정됐다.

encode() 메서드는 다음과 같이 세 개의 인수를 받으며 payload를 암호화한다.

- **페이로드:** 값이 저장된 딕셔너리로, 인코딩할 대상이다.
- **키:** 페이로드를 사인하기 위한 키다.
- **알고리즘:** 페이로드를 사인 및 암호화하는 알고리즘으로, 기본값인 HS256 알고리즘이 가장 많이 사용된다.

애플리케이션에 전달된 토큰을 검증하는 함수를 jwt_handler.py에 추가해보자.

```
def verify_access_token(token: str):
    try:
        data = jwt.decode(token, settings.SECRET_KEY, algorithms=["HS256"])
```