```python
async def introspect_token(token: str, token_type_hint: str = "access_token") -> dict:
    """Introspects the token using the authorization server's introspection endpoint."""
    async with httpx.AsyncClient() as client:
        response = await client.post(
            INTROSPECTION_ENDPOINT,
            data={
                "token": token,
                "token_type_hint": token_type_hint,
                "client_id": CLIENT_ID,  # Include the client_id in the request payload
            },
            headers={
                "Content-Type": "application/x-www-form-urlencoded",  # Match the curl request
            },
        )
        response.raise_for_status()
        return response.json()
```

Introspect endpoint (remote validation w/IdP)

```python
if USE_INTROSPECTION:
    # Use introspection to validate the token
    introspection_result = await introspect_token(token)
    if not introspection_result.get("active"):
        raise HTTPException(
            status_code=status.HTTP_401_UNAUTHORIZED, detail="Token is inactive or invalid"
        )
    # Extract user data and scopes from introspection result
    user_data = {
        "sub": introspection_result["sub"],
        "username": introspection_result.get("username", "unknown"),
    }
    token_scopes = introspection_result.get("scope", "").split()

    # Validate the 'aud' claim
    if introspection_result.get("aud") != "api://default":  # Replace with your expected audience
        raise HTTPException(
            status_code=status.HTTP_401_UNAUTHORIZED, detail="Invalid audience"
        )
```