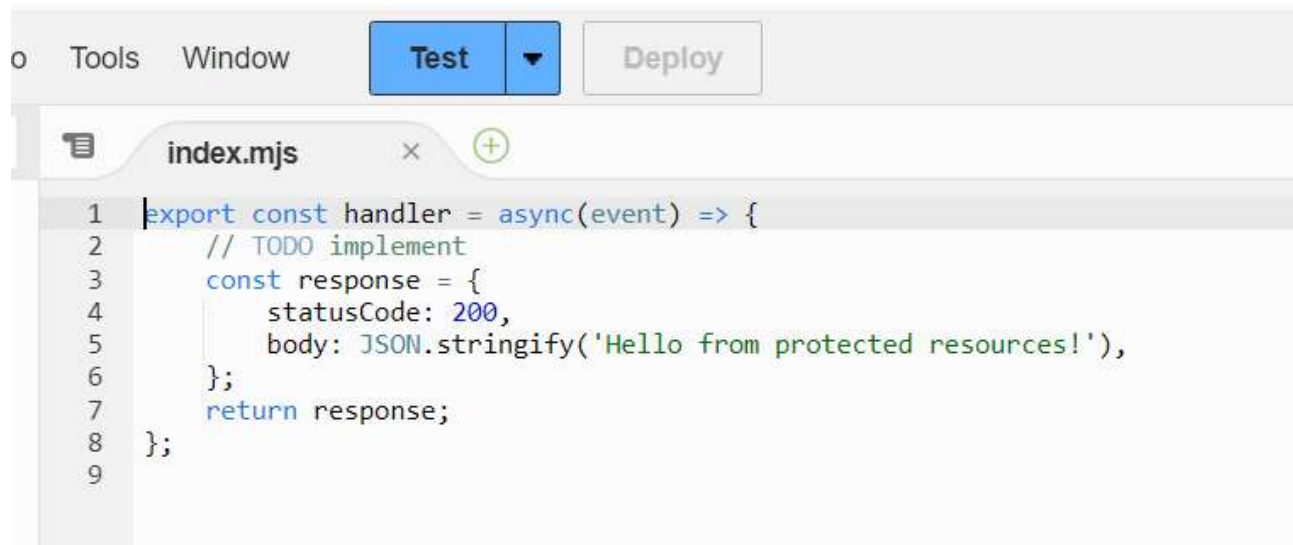


Cognito와 API gateway 동작 원리 탐구

API gateway에서 인증이 필요한 라우팅을 지정하고 해당 인증을
Cognito에서 제공하는 JWT를 이용하기

출력을 내보낼 Lambda 함수 생성

- 본 Lambda 함수는 Spring 등 어플리케이션을 대신한다.



The screenshot shows a code editor window with a menu bar containing 'Tools' and 'Window'. Below the menu bar are two buttons: 'Test' (highlighted in blue) and 'Deploy'. The editor displays a file named 'index.mjs' with the following JavaScript code:

```
1 export const handler = async(event) => {
2   // TODO implement
3   const response = {
4     statusCode: 200,
5     body: JSON.stringify('Hello from protected resources!'),
6   };
7   return response;
8 };
9
```

라우팅에 사용할 API gateway 생성 및 테스트

- API gateway를 Lambda와 연결하여 테스트 한다. 문구 정상 출력

pets-api

API 세부 정보

API ID	프로토콜
vu01ef4se1	HTTP
설명	기본 엔드포인트
to test integration with cognito	활성화됨

pets-api에 대한 스테이지

리소스 찾기

스테이지 이름	URL 호출
\$default	https://vu01ef4se1.execute-api.ap-northeast-2.amazonaws.com

pets-api에 대한 경로

검색

▼ /pets

ANY AWS Lambda

GET

https://vu01ef4se1.execute-api.ap-northeast-2.amazonaws.com/pets

Body Cookies Headers (5) Test Results

Pretty

Raw

Preview

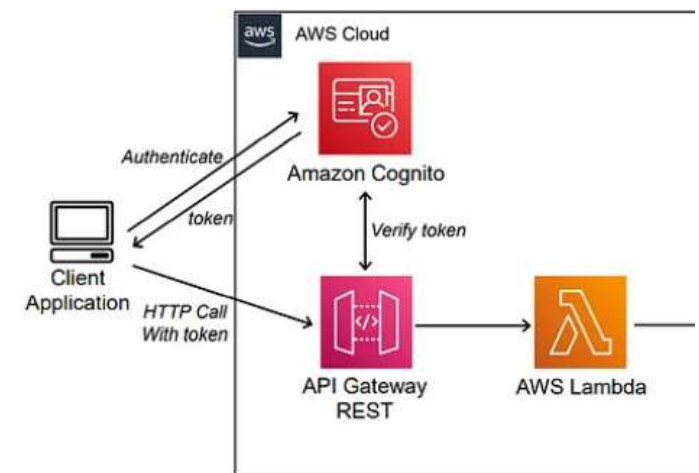
Visualize

Text

1 "Hello from protected resources!"

Cognito의 역할

- Cognito는 로그인에 성공하면 redirect 시키면서 쿼리 스트링에 인증 관련 내용을 첨부해 준다.
- 인증 내용을 JWT 형태로 보낼 수 있다. JWT 형태로 보내면 HTTP API gateway와 통합할 수 있다.
- HTTP API gateway에는 JWT 권한 부여자를 사용하여 쿼리 스트링의 JWT를 인식하고 이를 Cognito에 보내서 알맞은 토큰인지 확인한다.
- 알맞은 토큰일 경우 뒤에 어플리케이션 코드를 호출하여 진행한다.



Cognito 사용자 풀 생성

- 다른 설정은 크게 중요한 게 없으므로 그대로 진행해도 무방
- 앱통합에서 호스팅 UI를 생성해야 한다.

☒ Cognito 호스팅 UI 사용

Amazon Cognito에서 호스팅 방식의 가입, 로그인 및 OAuth 2.0 서비스 엔드포인트를 구축합니다. 이 기능을 사용하지 않는 API 작업을 사용하여 가입 및 로그인을 수행합니다.

도메인 정보

호스팅 UI 및 OAuth 2.0 엔드포인트에 대한 도메인을 구성합니다. 호스팅 UI를 사용하려면 인증 엔드포인트를 생성할 도메인을 선택합니다.

도메인 유형

☒ Cognito 도메인 사용

Amazon 소유 도메인에서 사용할 식별 접두사를 입력합니다. 프로덕션 앱의 경우 사용자 지정 도메인을 대신 사용하는 것이 좋습니다.

☐ 사용자 지정 도메인 사용

Cognito 호스팅 가입 및 로그인 페이지에 대해 소유한 도메인을 입력합니다. 사용자 지정 도메인을 사용하려면 DNS 레코드와 Certificate Manager(ACM) 인증서를 제공해야 합니다. 프로덕션 워크로드에는 사용자 지정 도메인을 사용하는 것이 좋습니다.

Cognito 도메인

도메인 접두사를 입력합니다.

.auth.ap-northeast-2.amazoncognito.com

Cognito 사용자 풀 생성

- 허용된 콜백 URL은 로그인 후 돌아갈 주소 리스트

허용된 콜백 URL [정보](#)

인증 후 사용자를 다시 리디렉션할 콜백 URL을 하나 이상 입력합니다. 이 URL은 일반적으로 Cognito에서 발현의 URL입니다. HTTPS URL과 사용자 지정 URL 스키마를 사용할 수 있습니다.

URL

콜백 URL의 길이는 1~1024자여야 합니다. 유효한 문자는 문자, 마크, 숫자, 기호 및 구두점입니다. Amazon Cognito는 테스트 목적으로만 `http://localhost` 를 제외하고 HTTP를 통한 HTTPS를 필요로 합니다. `myapp://example`와 같은 앱 콜백 URL도 지원됩니다. 조각을 포함하면 안 됩니다.

Cognito 사용자 풀 생성 (핵심!)

- 가장 핵심은 고급 앱 클라이언트 설정에서 코드 권한 체크를 제거하고 암시적 권한 부여를 선택해야 한다는 것이다.
- 이걸 체크해야 JWT를 발급받을 수 있다.

자격 증명 공급자 [정보](#)

이 앱 클라이언트에 사용할 수 있는 자격 증명 공급자를 선택합니다.

자격 증명 공급자 선택

Cognito 사용자 풀

사용자는 이메일, 전화번호 또는 사용자 이름을 사용하여 Cognito에 로그인할 수 있습니다.

OAuth 2.0 권한 부여 유형 [정보](#)

OAuth 권한 부여 유형을 하나 이상 선택하여 Cognito가 이 앱에 토큰을 전달하는 방법을 구성합니다. 선택한 앱 유형에 따라.

OAuth 2.0 권한 부여 유형 선택

암시적 권한 부여

클라이언트가 액세스 토큰(그리고 범위에 따라 선택적으로 ID 토큰)을 직접 가져와야 함을 지정합니다.

API gateway에서 권한 부여자 설정

- 권한 부여자 유형에서 JWT를 선택하면 JWT를 발급하도록 설정해놓은 Cognito를 이용하여 인증 절차를 진행할 수 있다.

권한 부여자 생성

권한 부여자 유형 [정보](#)



JWT

JSON Web Token(JWT)을 사용하여 API에 대한 액세스를 제어합니다.

If you want to attach IAM authorization, go back to [Attach authorizers to routes](#) and select through a built-in authorizer in the console.

API gateway에서 권한 부여자 설정

- 발급자 URL 형식은 아래처럼 따라하고 마지막에 사용자 풀 ID를 넣는다.
- 대상에는 앱 클라이언트의 ID를 넣는다.

자격 증명 소스

토큰의 소스를 정의하는 선택 표현식을 입력하십시오.

`$request.header.Authorization`

발급자 URL

일반적으로 권한 부여 서버의 잘 알려진 메타데이터 엔드포인트의 발급자 필드에 있는 자격 증명 공급자의 발급자 URL을 입력하십시오.

`https://cognito-idp.ap-northeast-2.amazonaws.com/ap-northeast-2_4IHft1wah`

대상

자격 증명 공급자에 등록된 클라이언트 ID 또는 권한 부여자가 확인해야 하는 JWT 대상 클레임의 임의의 문자열을 입력하십시오.

`4ok6hfd2cngtfar6rh6vc`

제거

사용자 풀 이름

`pets-pool`

사용자 풀 ID

`ap-northeast-2_4IHft1wah`

앱 클라이언트 이름

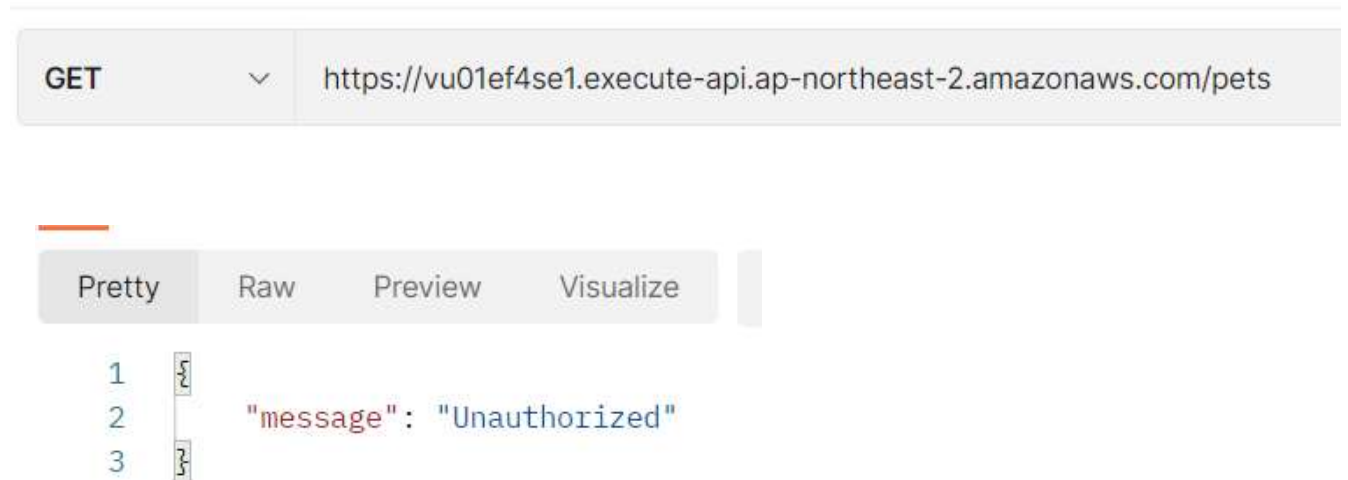
`pets-client`

클라이언트 ID

`4ok6hfd2cngtfar6rh6vcvbcsf`

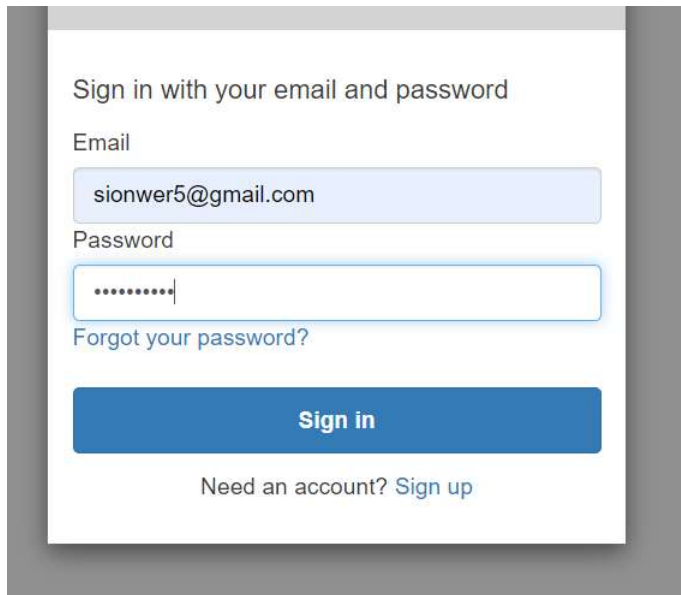
API gateway에서 권한 부여 테스트

- 아무런 헤더 없이 접근하면 Unauthorized 가 뜬다.



Cognito 앱 클라이언트 호스팅 UI에서 로그인

- 호스팅 UI에서 로그인을 실행하면 콜백 URI로 리다이렉트 된다.
- 리다이렉트된 주소에는 쿼리스트링에 JWT가 포함되어있다.



Sign in with your email and password

Email

sionwer5@gmail.com

Password

.....

[Forgot your password?](#)

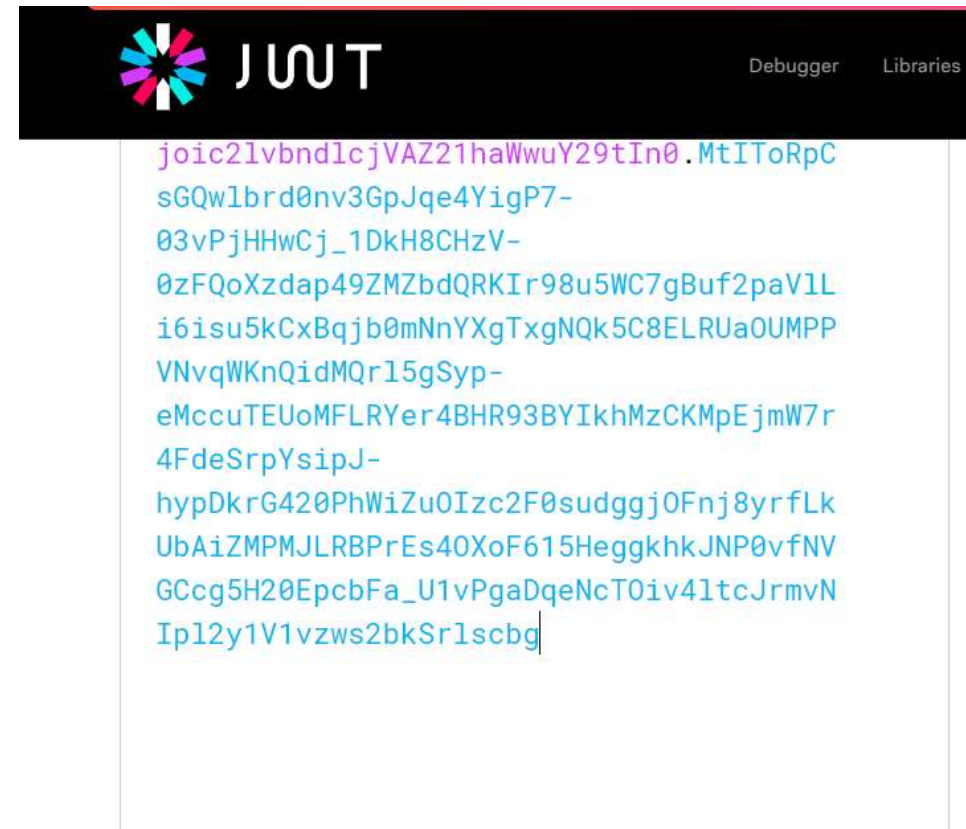
Sign in

[Need an account? Sign up](#)

localhost:3000/#id_token=eyJraWQiOiI4K29rNjY4VnFHTGZEMFwwVUhtcEpxZjBYckxjV3hxTXpONTRoT2JZbUNqQT0iLCJhbGciOiJS

JWT 내용 정상 판별

- 쿼리스트링의 JWT에서 id_token 내용만 decode 하면 정상적인 JWT로 판별이 난다.
- access_token은 빼고 decode해야 한다.



✔ Signature Verified

JWT를 Authorization 헤더에 끼워 넣고 요청

- 권한으로 Blocking된 URI를 이제는 JWT를 헤더에 끼워넣고 요청을 보낸다.
- 헤더 이름은 Authorization, Id_token 내용만 가져와서 값에 넣는다.

<input checked="" type="checkbox"/>	Authorization	eyJraWQiOiI4K29rNjY4VnFHTGZEMFwvVUhtcEp...
	Key	Value

- JWT를 함께 보내면 해당 주소로 정상 진입하는 것을 확인할 수 있다.

Body Cookies Headers (5) Test Results

Pretty

Raw

Preview

Visualize

Text



1 "Hello from protected resources!"

