

🌟 Language Generator Function 만들기

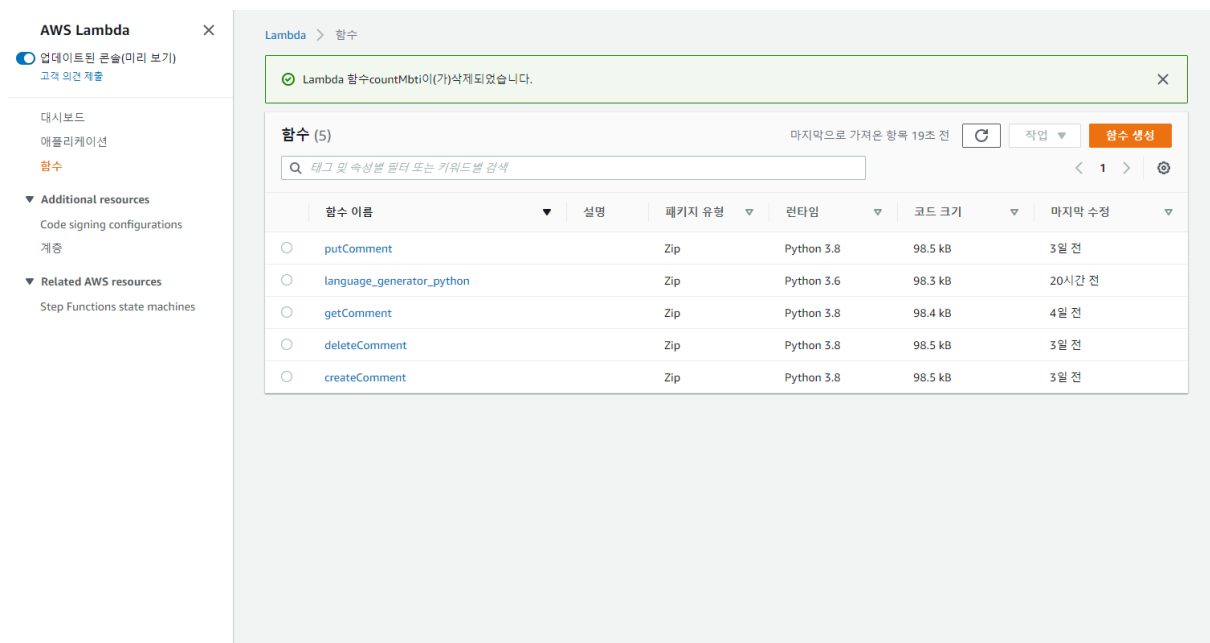
목적: AWS Lambda와 API Gateway를 활용하여 랜덤 닉네임 생성기를 만듭니다.

제작자: 송은주

준비물: AWS 계정, Nouns 정보가 담겨있는 RDS(MySQL) 정보, lan.zip(lambda에 넣을 module이지만, 없으면 직접 만드시면 됩니다.)

1. 먼저 Lambda function 만들기

AWS Lambda 콘솔 로 들어와주세요.



The screenshot shows the AWS Lambda console interface. On the left, there's a sidebar with navigation options like 'AWS Lambda', '업데이트된 콘솔(미리 보기)', '고객 의견 제출', '대시보드', '애플리케이션', '함수', 'Additional resources', and 'Related AWS resources'. The main area displays a list of Lambda functions under the heading '함수 (5)'. A green notification banner at the top states 'Lambda 함수countMbtI(가)삭제되었습니다.' Below the notification, there's a search bar and a table of functions. The table has columns for '함수 이름', '설명', '패키지 유형', '런타임', '코드 크기', and '마지막 수정'.

함수 이름	설명	패키지 유형	런타임	코드 크기	마지막 수정
putComment		Zip	Python 3.8	98.5 kB	3일 전
language_generator_python		Zip	Python 3.6	98.3 kB	20시간 전
getComment		Zip	Python 3.8	98.4 kB	4일 전
deleteComment		Zip	Python 3.8	98.5 kB	3일 전
createComment		Zip	Python 3.8	98.5 kB	3일 전

오른쪽 위의 함수 생성 을 눌러 주세요.

Lambda > 함수 > 함수 생성

함수 생성 Info

다음 옵션 중 하나를 선택하여 함수를 생성합니다.

새로 작성

간단한 Hello World 예제는 시작하십시오.

블루프린트 사용

샘플 코드 및 구축 Lambda 애플리케이션을 위한 구성 사전 설정을 일반적인 사용 사례를 살펴봅니다.

컨테이너 이미지

함수에 대해 배포할 컨테이너 이미지를 선택합니다.

서버리스 앱은 리포지토리 찾아 보기

샘플 Lambda 애플리케이션을 배포하십시오. AWS Serverless Application Repository

기본 정보

함수 이름
함수의 용도를 설명하는 이름을 입력합니다.

countMbti

공백 없이 문자, 숫자, 하이픈 또는 밑줄만 사용합니다.

런타임 Info
함수를 작성할 때 사용할 언어를 선택합니다.

Python 3.8

권한 Info
기본적으로 Lambda는 Amazon CloudWatch Logs에 로그를 업로드하는 권한을 가진 실행 역할을 생성합니다. 이 기본 역할은 나중에 트리거를 추가할 때 사용자 지정할 수 있습니다.

▼ 기본 실행 역할 변경

실행 역할
함수에 대한 권한을 정의하는 역할을 선택합니다. 사용자 지정 역할을 생성하려면 [IAM 콘솔](#)로 이동하십시오.

☒ 기본 Lambda 권한을 가진 새 역할 생성
☐ 기존 역할 사용
☐ AWS 정책 템플릿에서 새 역할 생성

함수 이름을 작성해 주세요. (원하시는 대로 작성하시면 됩니다. 저는 다른 함수를 시험삼아 작성할 때 캡쳐해 놔던 것이라서 이름이 뜬금없이 countMbti 입니다. 적절한 이름으로 만들어주세요.) 런타임은 Python 3.8 으로 선택해 주세요.

Lambda > 함수 > countMbti

countMbti

조절 ARN 복사 작업 ▼

▼ 함수 개요 Info

countMbti
Layers (0)

+ 트리거 추가 + 대상 추가

설명

-

마지막 수정

2초 전

함수 ARN

arn:aws:lambda:ap-northeast-2:408170130409:function:countMbti

코드 테스트 모니터링 구성 별칭 버전

코드 소스 Info 에서 업로드 ▼

File Edit Find View Go Tools Window Test Deploy Changes deployed

Go to Anything (Ctrl-P)

Environment

- countMbti /
- lambda_function.py

Execution results

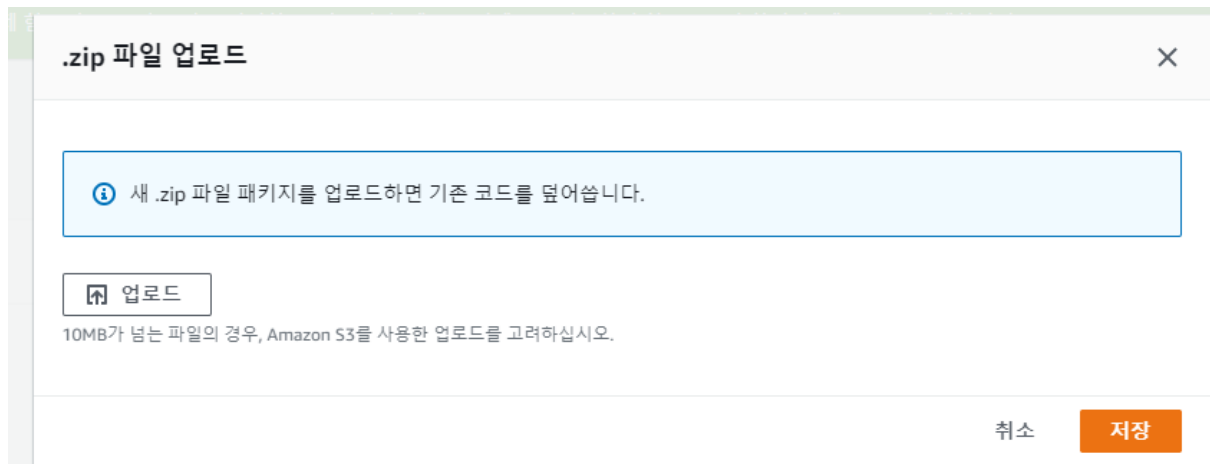
No execution results yet!

만들고 난 후 사진입니다.

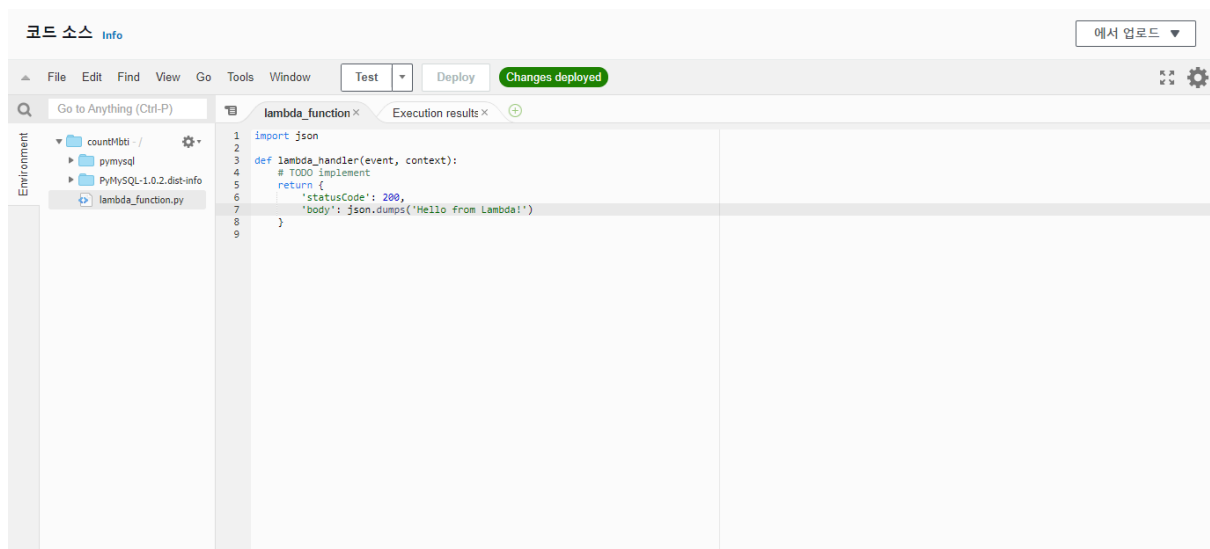
Lambda function 에서 어떻게 모듈을 import해서 써올 수 있을까요?

정답은 로컬 에서 빈 폴더에 원하는 모듈을 가져다 놓고, zip 파일 형식으로 업로드하게되면 사용할 수 있습니다. 여기서 이제 준비물 lan.zip 이 필요합니다. (RDS를 쓰기 위한

`pymysql` 이 들어가 있는 압축 파일입니다.) 오른쪽의 `에서 업로드` 를 누르고 `.zip 파일 업로드` 를 해줍니다.



`lan.zip` 을 올리게 되면 아래와 같은 코드 소스가 보이게 될 거예요. 참고로 `Lambda function` 은 `python` 기준으로는 `lambda_function.py` 라는 함수가 있고, 그 함수 안에 리턴 값으로 `statusCode` 가 꼭! 들어가야 해요. `lan.zip` 파일이 없으신 분들은 아래처럼 `pymysql` 을 가져오셔서 저 모양대로 파일을 꾸리고 압축해서 올리시면 됩니다.



잘 입력이 되었나 확인하기 위해 `Test` 를 해볼 예정입니다. `Deploy` 옆의 `Test` 버튼을 눌러주세요.

테스트 이벤트 구성



함수는 최대 10개의 테스트 이벤트를 가질 수 있습니다. 이 이벤트는 지속되므로 다른 컴퓨터 또는 웹 브라우저로 전환하고 동일한 이벤트로 함수를 테스트할 수 있습니다.

☒ 새로운 테스트 이벤트 생성

☐ 저장된 테스트 이벤트 편집

이벤트 템플릿

hello-world

이벤트 이름

MyEventName

```
1 {
2   "key1": "value1",
3   "key2": "value2",
4   "key3": "value3"
5 }
```

이벤트 이름을 원하시는 대로 작성해주시고, 아래의 `json` 파일은, 그러니까, `http`로 따지자면 `QueryString`을 어떻게 넣어서 보내줄 지 테스트하는 과정입니다. 지금은 그대로 설정하고 이벤트 생성 합시다.

코드 소스 Info

File Edit Find View Go Tools Window Test Deploy Changes deployed

Go to Anything (Ctrl-P)

Environment

- countMbt - /
 - pymysql
 - PyMySQL-1.0.2.dist-info
 - lambda_function.py

Execution results

Status: Succeeded | Max memory used: 51 MB | Time: 1.24 ms

Response

```
{
  "statusCode": 200,
  "body": "\\Hello from Lambda!"
}
```

Function Logs

```
START RequestId: 9eae7a2-3792-42cf-95de-f6fa6fffd4a11 Version: $LATEST
END RequestId: 9eae7a2-3792-42cf-95de-f6fa6fffd4a11
REPORT RequestId: 9eae7a2-3792-42cf-95de-f6fa6fffd4a11 Duration: 1.24 ms Billed Duration: 2 ms Memory Size: 128 MB Max Memory Used: 51 MB Init Duration: 129.46 ms
```

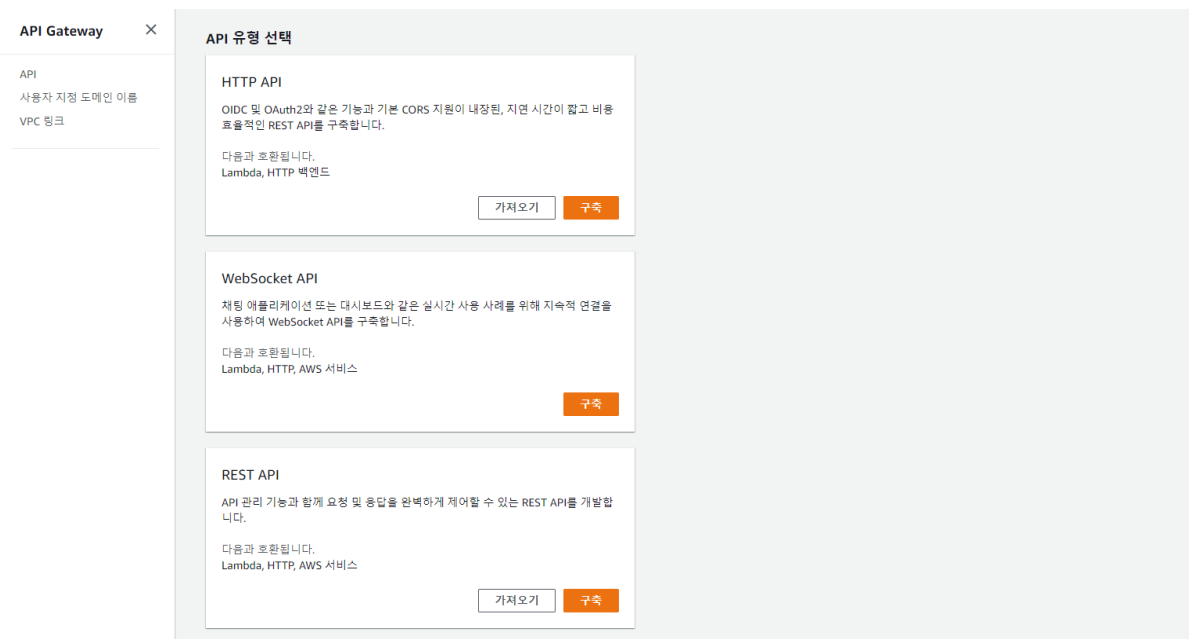
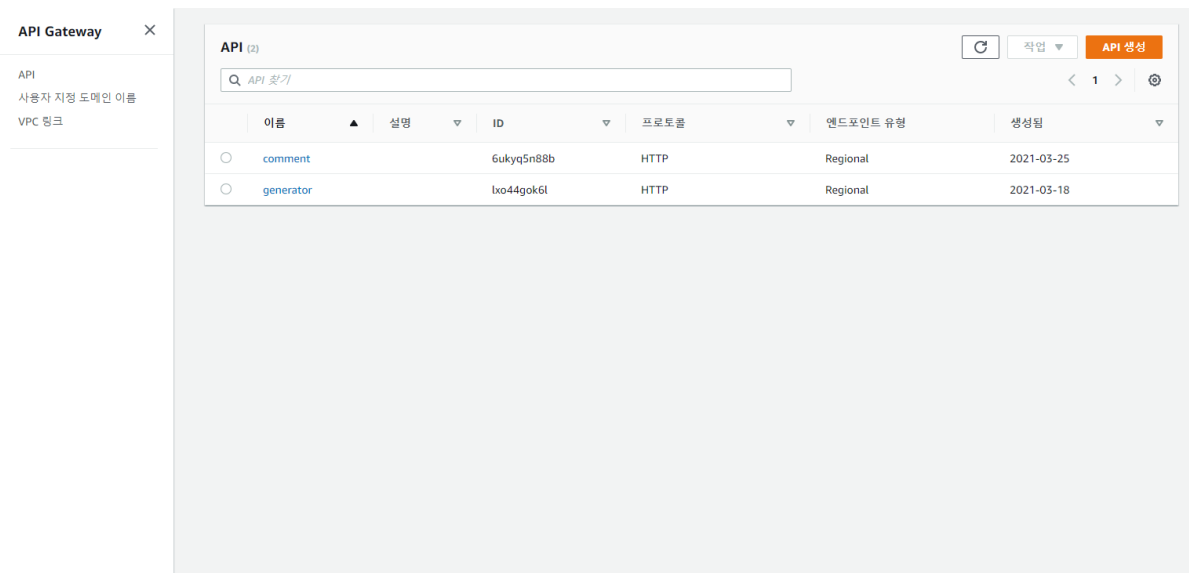
Request ID

9eae7a2-3792-42cf-95de-f6fa6fffd4a11

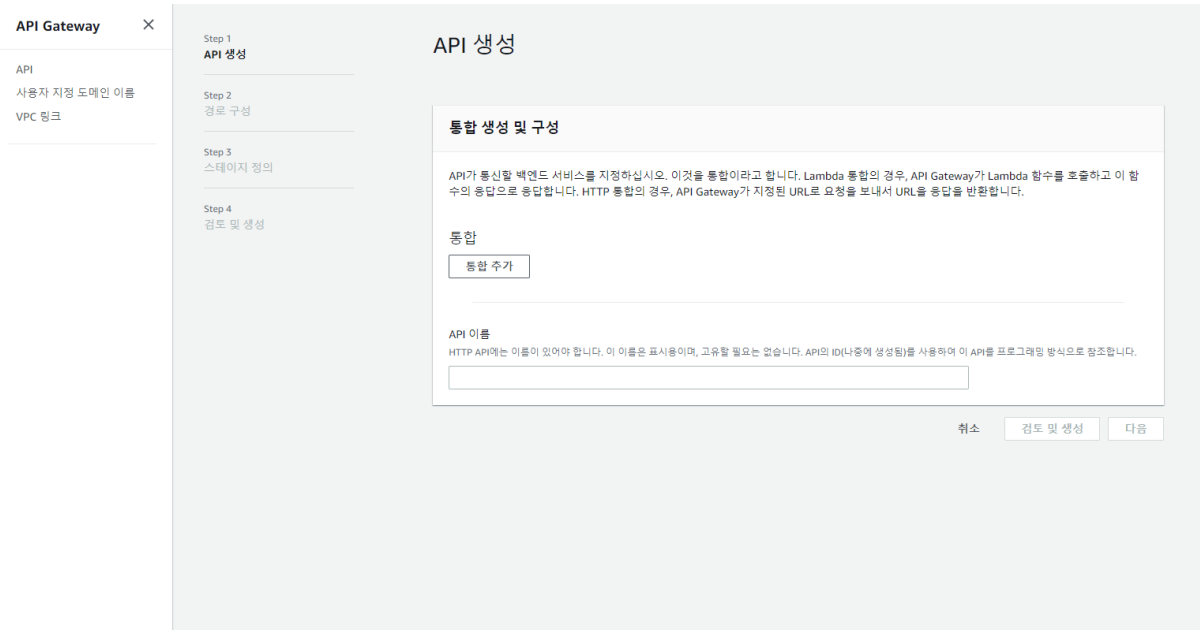
함수를 **Deploy** 를 해주시고, **Test** 버튼을 눌렀을 때 **Response** 가 오면 제대로 만들어 졌습니다.

2. API Gateway 만들고 Lambda와 연결하기

AWS API Gateway 콘솔 로 들어가세요. 오른쪽의 **API 생성** 을 누르세요.



HTTP API 의 **구축** 버튼을 눌러줍니다.



통합 추가 를 눌러주세요.



Lambda 를 눌러주세요.

통합 생성 및 구성

API가 통신할 백엔드 서비스를 지정하십시오. 이것을 통합이라고 합니다. Lambda 통합의 경우, API Gateway가 Lambda 함수를 호출하고 이 함수의 응답으로 응답합니다. HTTP 통합의 경우, API Gateway가 지정된 URL로 요청을 보내서 URL을 응답을 반환합니다.

통합

Lambda ▼

제거

AWS 리전

Lambda 함수

버전 자세히 알아보기.

ap-northeast-2 ▼

X

2.0 ▼

통합 추가

API 이름

HTTP API에는 이름이 있어야 합니다. 이 이름은 표시용이며, 고유할 필요는 없습니다. API의 ID(나중에 생성됨)를 사용하여 이 API를 프로그래밍 방식으로 참조합니다.

[취소](#)[검토 및 생성](#)[다음](#)

작성하셨던 **리전** 을 선택하시고 **돋보기** 모양을 누르시면 아까 만드셨던 **Lambda** 함수가 나옵니다. 저는 아까 **countMbti** 라고 만들었기 때문에 이를 선택하였습니다. **API 이름** 을 작성하시고 **다음** 을 누르세요.

경로 구성

API Gateway는 경로를 사용하여 API 사용자에게 통합을 표시합니다. HTTP API에 대한 경로는 HTTP 메서드와 리소스 경로(예: GET /pets)의 두 부분으로 구성됩니다. 통합에 대한 특정 HTTP 메서드(GET, POST, PUT, PATCH, HEAD, OPTIONS 및 DELETE)를 정의하거나 ANY 메서드를 사용하여 지정된 리소스에서 정의하지 않은 모든 메서드를 일치시킬 수 있습니다.

메서드

리소스 경로

통합 대상

GET ▼

/countMbti

→

countMbti ▼

제거

경로 추가

[취소](#)[이전](#)[다음](#)

메서드 는 어떤 메서드를 받을 지 판단합니다. **리소스 경로** 는 어떤 **endpoint url** 로 올지 결정합니다. 만드시려고 하시는 함수마다 적절하게 작성해 주시면 됩니다.

검토 및 생성

API 이름 및 통합

API 이름

countMbti

통합

countMbti (Lambda)

편집

경로

경로

GET /countMbti → countMbti (Lambda)

편집

Stages

스태이지

\$default (Auto-deploy: enabled)

편집

취소

이전

생성

최종본입니다. 다르게 설정하신 부분이 있다면 다르게 나오겠죠!

API Gateway

API

사용자 지정 도메인 이름

VPC 링크

API: countMbti... (djhtbgnrxl)

개발

경로

권한 부여

통합

ORS

다시 가져오기

내보내기

배포

스태이지

보호

조절 중

모니터링

지표

로그

API countMbti... 생성 완료

API Gateway > 세부 정보

스테이지: - 배포

countMbti

API 세부 정보

API ID

프로토콜

생성됨

djhtbgnrxl

HTTP

2021-03-29

설명

기본 엔드포인트

활성화됨

No Description

countMbti에 대한 스테이지

Q 리소스 찾기

스테이지 이름	URL 호출	연결된 배포	자동 배포	마지막 업데이트
\$default	https://[redacted].execute-api.ap-northeast-2.amazonaws.com	whzhba	enabled	2021-03-29

태그 (0)

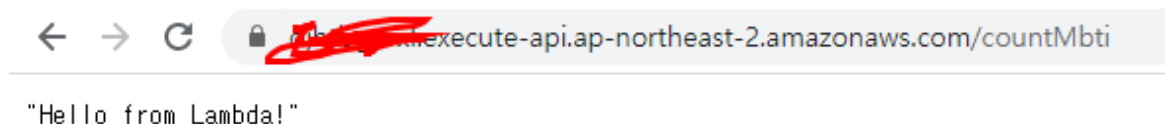
Q 리소스 찾기

키

값

태그 없음

만들고 나시면 위의 사진처럼 생성이 되게 됩니다. 중간의 **URL 호출** 부분을 복사하시고, 뒤에 아까 설정하셨던 **endPoint** 를 작성해주세요.



저로 따지면 위의 **URL** 을 복사한 뒤, 뒤에 **/countMbti** 를 붙이면 되겠죠!

그러면 결과가 아까 Lambda의 **Hello from Lambda** 가 나와야 합니다. 나오면 **API Gateway** 와 **Lambda** 연결 성공입니다.

한 **API Gateway** 에 1 개의 **Lambda** 만 연결할 수 있는 것은 아닙니다. **API Gateway 콘솔** 에서 **개** **발-경로** 로 들어가시면 만들었던 **Lambda** 를 계속해서 넣어줄 수 있습니다.

3. 아까 만들었던 Lambda를 수정해서 닉네임 생성기 만들기

1번 에서 만들었던 **Lambda 함수** 를 수정합니다.

```
# -- coding: utf-8 --
import json, pymysql, random

def lambda_handler(event, context):
    MIN = 1
    MAX = 25966 + 1
    conn = pymysql.connect(host="HOSTNAME주소", port=3306, user="USER이름",
        password="RDS비번", db="DB이름", charset="utf8")
    curs = conn.cursor()
    ranNum = random.sample(range(MIN, MAX), 2)

    sqlA = f"select nouns from nouns where id={ranNum[0]}"
    curs.execute(sqlA)
    tempA = curs.fetchall()

    sqlB = f"select nouns from nouns where id={ranNum[1]}"
    curs.execute(sqlB)
    tempB = curs.fetchall()

    conn.close()
    result = tempA[0][0] + "의 " + tempB[0][0]

    return {
        'headers': {
            'Access-Control-Allow-Headers': 'Content-Type',
            'Access-Control-Allow-Origin': '*',
```

```

    'Access-Control-Allow-Methods': 'OPTIONS,POST,GET'
  },
  'statusCode': 200,
  'body': json.dumps(result, ensure_ascii=False)
}

```

함수를 이렇게 바꿔주세요. `connect` 매개변수는 `RDS` 에서 있는 것을 적절히 설정해주세요.
 이후 `Deploy` 를 하고 `Test` 를 누르면!

```

response
{
  "headers": {
    "Access-Control-Allow-Headers": "Content-Type",
    "Access-Control-Allow-Origin": "*",
    "Access-Control-Allow-Methods": "OPTIONS,POST,GET"
  },
  "statusCode": 200,
  "body": "\"원심의 면사방적\""
}

```

순서대로 잘 따라왔다면 `랜덤으로 닉네임이 생성` 되어야 합니다. 아까 주소로 다시 들어가게되면

← → ↻ 🔒 ~~https://execute-api.ap-northeast-2.amazonaws.com/language_generator~~

"작용권의 오호츠크해고기압"

등으로 계속 랜덤해서 나오게 됩니다. 다만 사전에 있는 명사이다보니 아주 철학적이고 난해하고 묘한 이름이 나옵니다.

그래서 뭐? 이거 어따 쓰라고?

우리 `SSBTI` 에는 유형별로 채팅방이 있습니다. 이 채팅을 참여하게 되면 랜덤으로 닉네임을 만들게 되는데, 이 때 사용합니다.

즉, `front`단에서 닉네임을 생성하는 `.local.env` 등의 파일에서 주소를 위의 주소로 바꿔주시면 되겠죠?!