

Lab2. Serverless Hello World

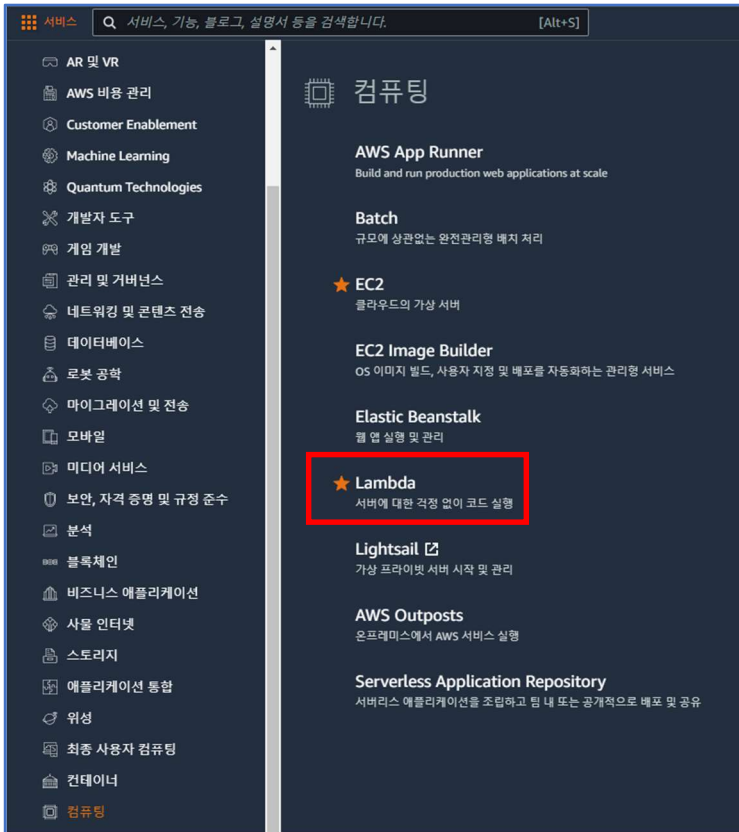
목적

Browser에 Hello World를 출력하는 초 간단 Serverless 학습하기

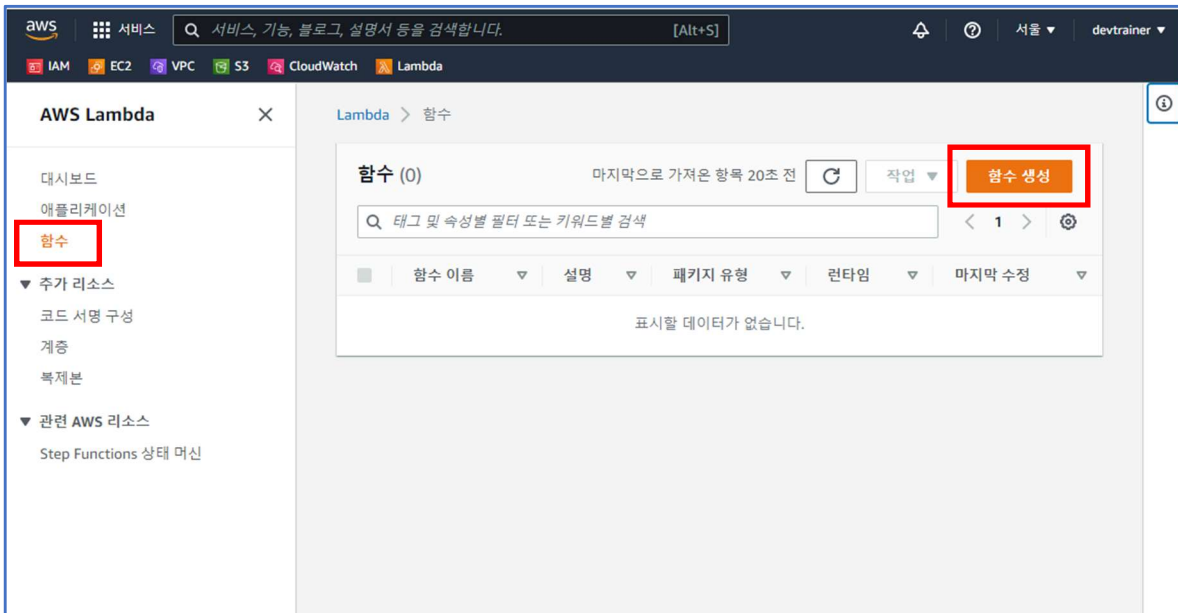
사전 준비물

AWS 계정

1. 로그인 후, [서비스] > [컴퓨팅] > [Lambda]를 클릭하여 **AWS Lambda** 페이지로 이동한다.



2. [AWS Lambda] 페이지에서 왼쪽 메뉴 중 [함수]를 선택한다. 반드시 현재 실습은 **서울 Region**에서 수행한다. 페이지 오른쪽 상단의 [함수 생성]을 클릭한다.



3. [함수 생성] 페이지에서 함수를 생성하기 위해 [새로 작성]을 선택한다.

Lambda > 함수 > 함수 생성

함수 생성 정보

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 찾아보기 ☐
샘플 Lambda 애플리케이션을 배포하십시오. AWS Serverless Application Repository

기본 정보

함수 이름
함수의 용도를 설명하는 이름을 입력합니다.

myFunctionName

공백 없이 문자, 숫자, 하이픈 또는 밑줄만 사용합니다.

4. [기본 정보] 섹션에서, [함수 이름]은 {계정이름}-HelloWorld-Python-Lambda, [런타임]은 Python 3.9를 선택하고, 페이지 오른쪽 하단의 [함수 생성]을 클릭한다. [권한] 같은 나머지 설정은 기본값을 사용하기로 한다.

기본 정보

함수 이름
함수의 용도를 설명하는 이름을 입력합니다.

Lab-HelloWorld-Python-Lambda

공백 없이 문자, 숫자, 하이픈 또는 밑줄만 사용합니다.

런타임 정보
함수를 작성하는 데 사용할 언어를 선택합니다. 콘솔 코드 편집기는 Node.js, Python 및 Ruby만 지원합니다.

Python 3.9

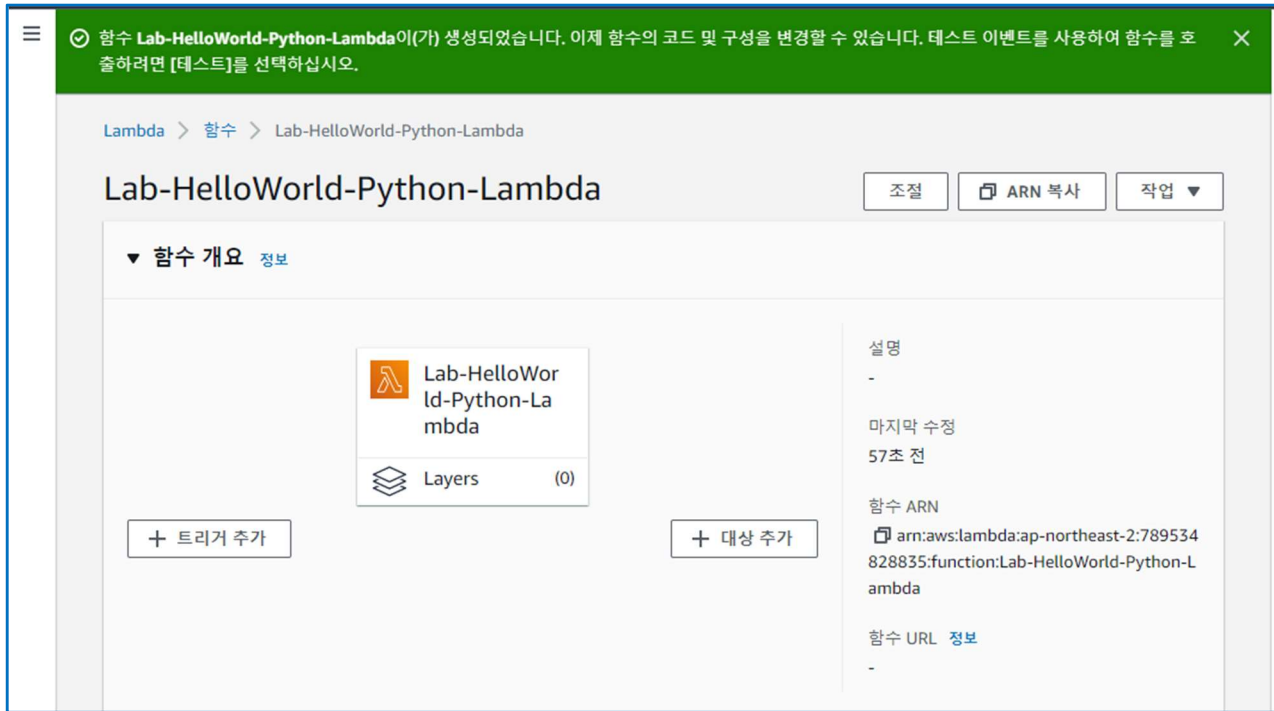
권한 정보
기본적으로 Lambda는 Amazon CloudWatch Logs에 로그를 업로드하는 권한을 가진 실행 역할을 생성합니다. 이 기본 역할은 나중에 트리거를 추가할 때 사용자 지정할 수 있습니다.

▶ 기본 실행 역할 변경

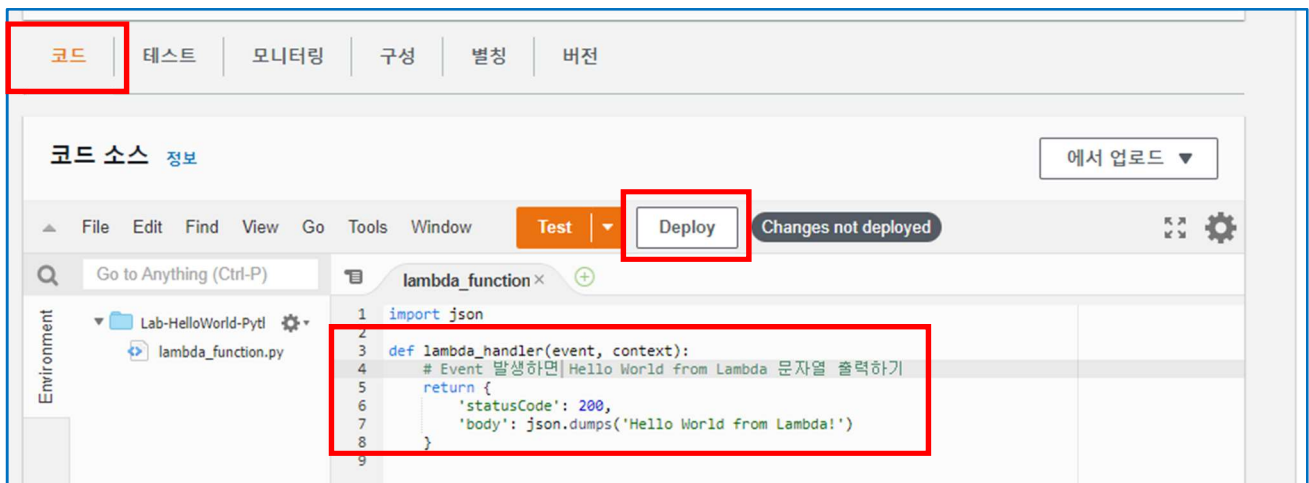
▶ 고급 설정

취소 **함수 생성**

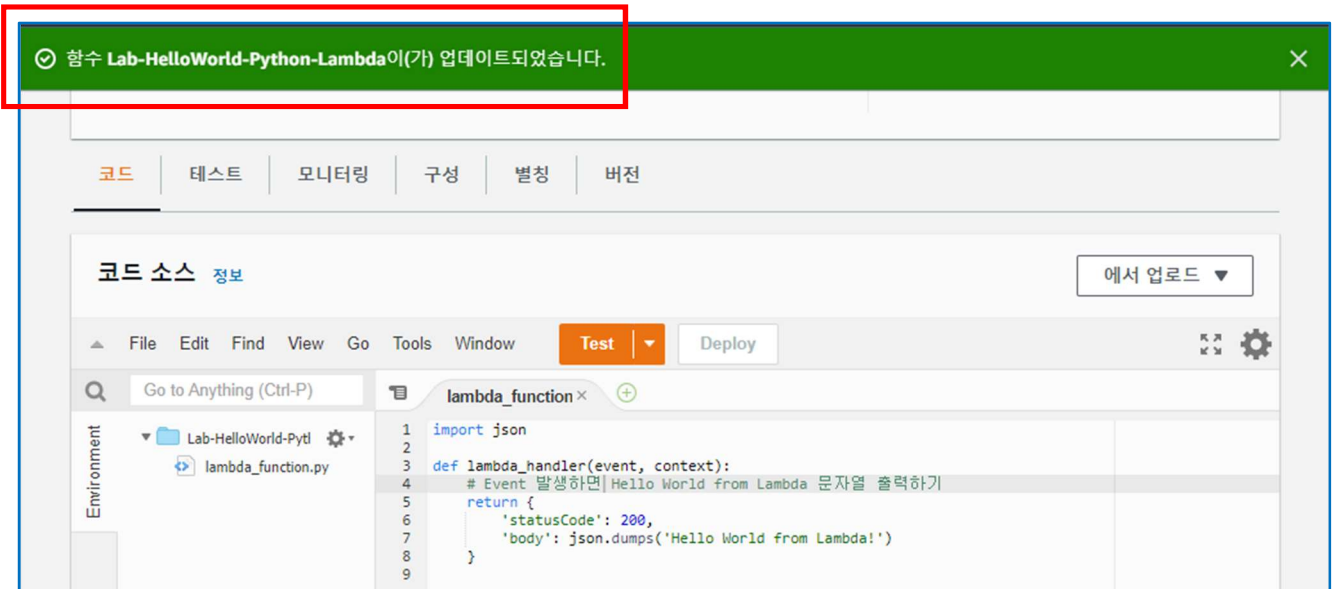
5. 정상적으로 **Lambda 함수**가 잘 생성되었다.



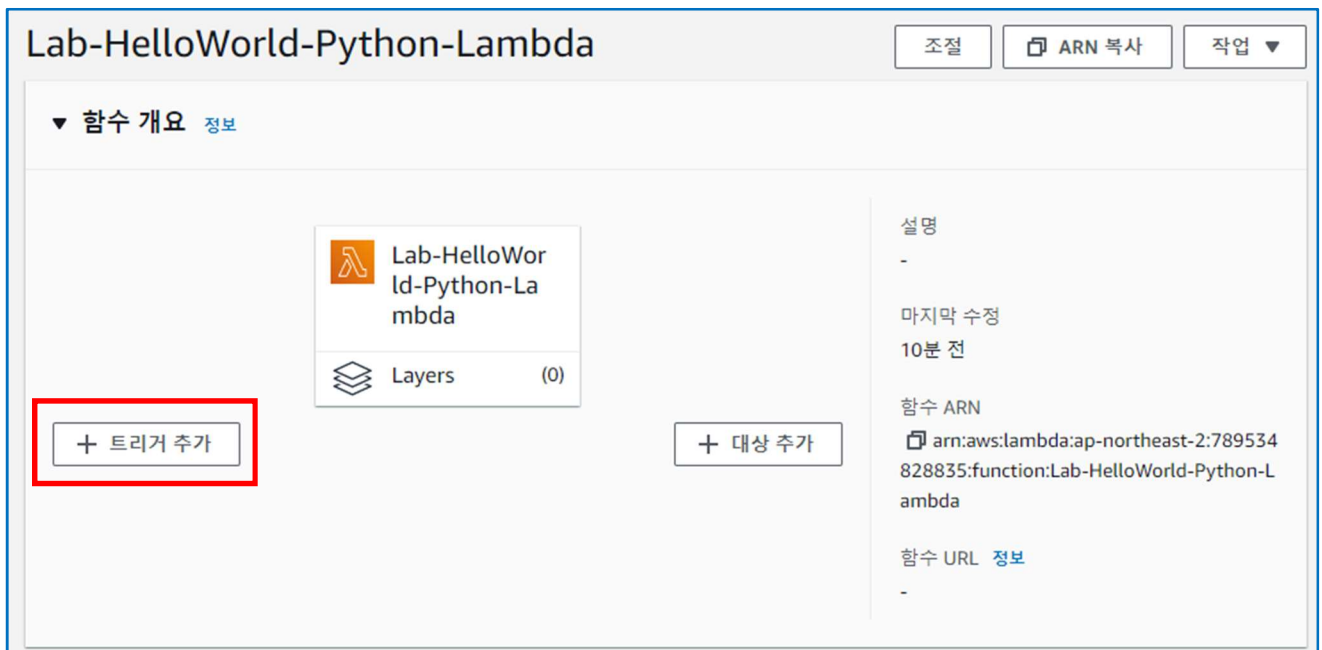
6. 페이지를 아래로 스크롤다운하여, **[코드]**탭에서 다음과 같이 코드를 간단히 수정한다. 수정이 완료되었으면 **[Deploy]** 버튼을 클릭하여 배포한다.



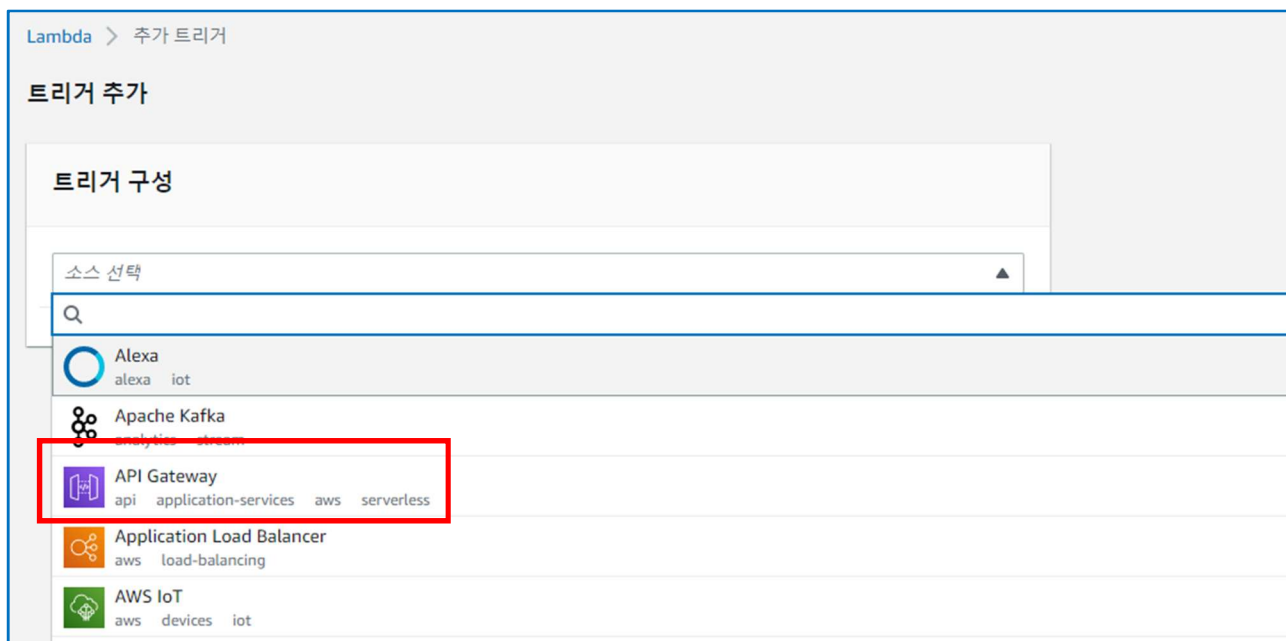
7. 생성한 코드가 잘 배포되었다.



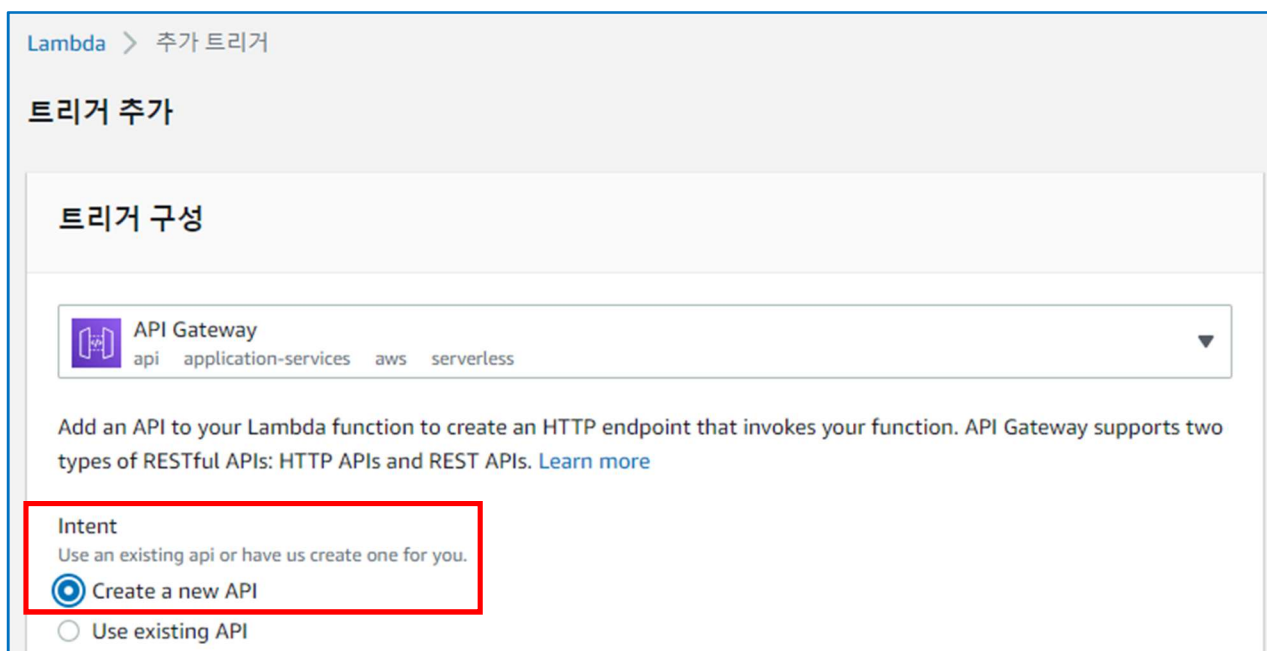
8. 배포가 완료되었으면 이 함수를 호출하게 하는 트리거를 추가해보자. 페이지를 다시 스크롤업하여 [트리거 추가]를 클릭한다.



9. [트리거 추가] 페이지이다. [트리거 구성] 섹션에서, [소스 선택] 드롭다운 목록에서 [API Gateway]를 선택한다.



10. [Intent]에서 새 API를 생성하기 위해 [Create a new API]를 선택한다.



11. **[API type]**은 기본값 **[HTTP API]**를 선택하고, **[Security]**은 **[Open]**을 선택하여 인터넷 주소만 안다면 누구나 인증키 없이 접근할 수 있도록 한다. 그 아래 **[Additional settings]**를 클릭해보자.

The screenshot shows the 'API type' and 'Security' configuration options. The 'API type' section has two radio buttons: 'HTTP API' (selected) and 'REST API'. The 'Security' section has a dropdown menu set to 'Open'. Below these is an 'Additional settings' section with a blue arrow icon and a text box. At the bottom right are '취소' (Cancel) and '추가' (Add) buttons.

API type

☒ HTTP API
Build low-latency and cost-effective REST APIs with built-in features such as OIDC and OAuth2, and native CORS support.

☐ REST API
Develop a REST API where you gain complete control over the request and response along with API management capabilities.

Security
Configure the security mechanism for your API endpoint.

Open

▶ Additional settings

Lambda는 Amazon API Gateway이(가) 이 트리거에서 Lambda 함수를 호출하는 데 필요한 권한을 추가합니다. Lambda 권한 모델에 대해 자세히 알아보기.

취소 추가

12. **[API name]**은 기본적으로 생성된 이름 그대로 사용하기로 하고, 나머지 설정 역시 기본값 그대로 사용하기로 한다. 페이지 우측 하단의 **[추가]** 버튼을 클릭한다 .

The screenshot shows the 'Additional settings' section. The 'API name' field is highlighted with a red box and contains the text 'Lab-HelloWorld-Python-Lambda-API'. Below it is the 'Deployment stage' field with the value 'default'. There are two unchecked checkboxes: 'Cross-origin resource sharing (CORS)' and 'Enable detailed metrics'. At the bottom right, the '추가' (Add) button is highlighted with a red box. The '취소' (Cancel) button is also visible.

▼ Additional settings

API name
Choose a name for your API. API names don't need to be unique.

Lab-HelloWorld-Python-Lambda-API

Deployment stage
The name of your API's deployment stage.

default

☐ Cross-origin resource sharing (CORS)
CORS is required to call your API from a webpage that isn't hosted on the same domain. This option enables cross-origin resource sharing (CORS) from any domain by adding the Access-Control-Allow-Origin header to all responses.

☐ Enable detailed metrics
Record usage metrics for API routes. Standard CloudWatch pricing applies.

Lambda는 Amazon API Gateway이(가) 이 트리거에서 Lambda 함수를 호출하는 데 필요한 권한을 추가합니다. Lambda 권한 모델에 대해 자세히 알아보기.

취소 추가

13. 이렇게 해서 [API Gateway]를 생성했다. 생성한 [API Gateway]의 [API endpoint] 링크를 클릭해보자.

The screenshot shows the AWS Lambda console interface. At the top, there's a section for the function 'Lab-HelloWorld-Python-Lambda' with a 'Layers' section showing '(0)' layers. Below this, there's an 'API Gateway' section with a '+ 트리거 추가' button. To the right, there's a '설명' (Description) section with '마지막 수정' (Last modified) at '16분 전' (16 minutes ago) and '함수 ARN' (Function ARN) as 'arn:aws:lambda:ap-northeast-2:789534828835:function:Lab-HelloWorld-Python-Lambda'. Below the ARN, there's a '함수 URL 정보' (Function URL information) section with a '-' sign.

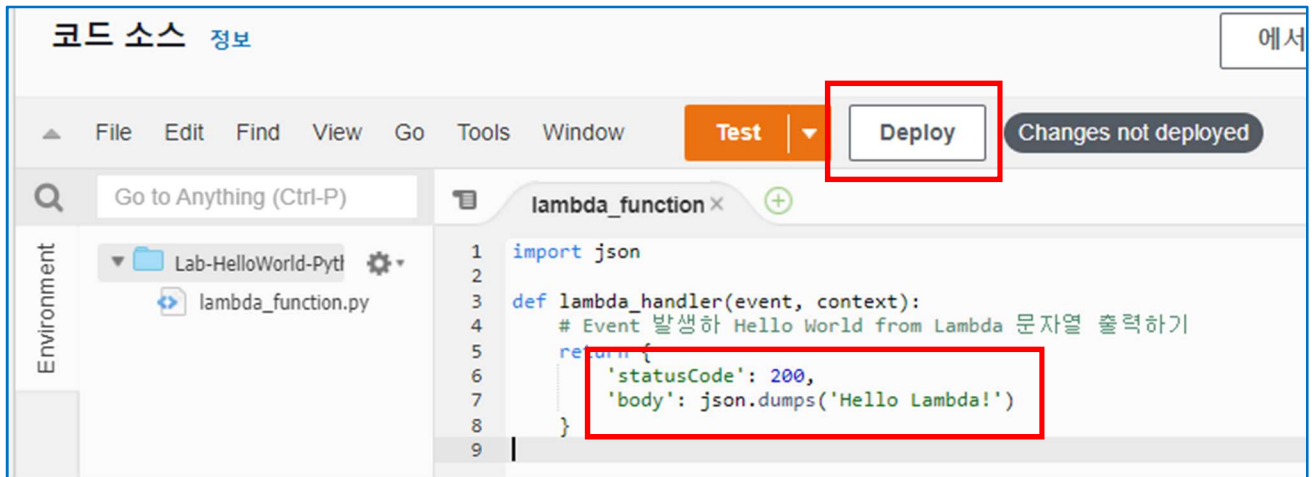
Below the function configuration, there's a navigation bar with tabs: '코드' (Code), '테스트' (Test), '모니터링' (Monitoring), '구성' (Configuration), '별칭' (Alias), and '버전' (Version). The '구성' (Configuration) tab is selected.

On the left, there's a sidebar with '일반 구성' (General configuration) and '트리거' (Triggers) sections. The '트리거' section is expanded, showing a list of triggers. One trigger is listed: 'API Gateway: Lab-HelloWorld-Python-Lambda-API' with the ARN 'arn:aws:execute-api:ap-northeast-2:789534828835:r0z7kdv53i/*/Lab-HelloWorld-Python-Lambda'. The 'API endpoint' is highlighted with a red box: 'https://r0z7kdv53i.execute-api.ap-northeast-2.amazonaws.com/default/Lab-HelloWorld-Python-Lambda'. Below the endpoint, there's a '세부 정보' (Details) link.

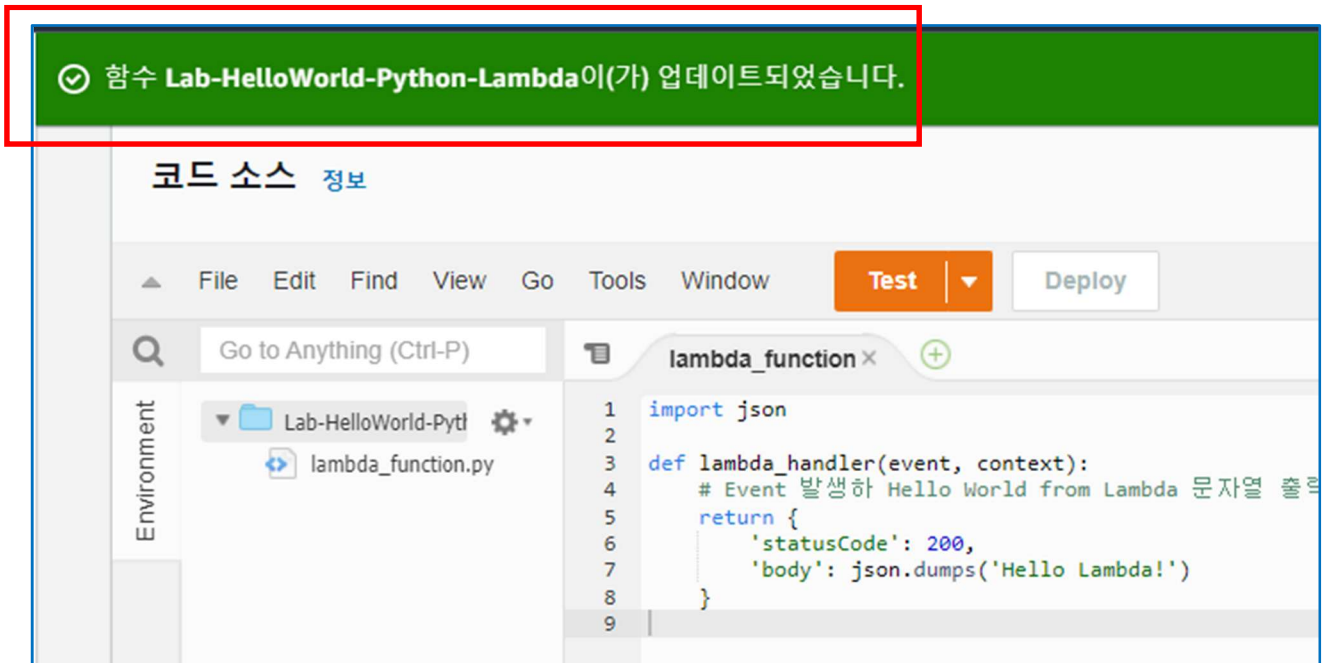
14. 앞에서 설정한 함수가 유저에게 전달하려고 했던 "Hello World from Lambda!"가 성공적으로 출력되는 것을 볼 수 있다.

The screenshot shows a web browser window. The address bar displays the URL 'https://r0z7kdv53i.execute-api.ap-northeast-2.amazonaws.com/default/Lab-HelloWorld-Python-Lambda'. The page content shows the text 'Hello World from Lambda!' highlighted with a red box.

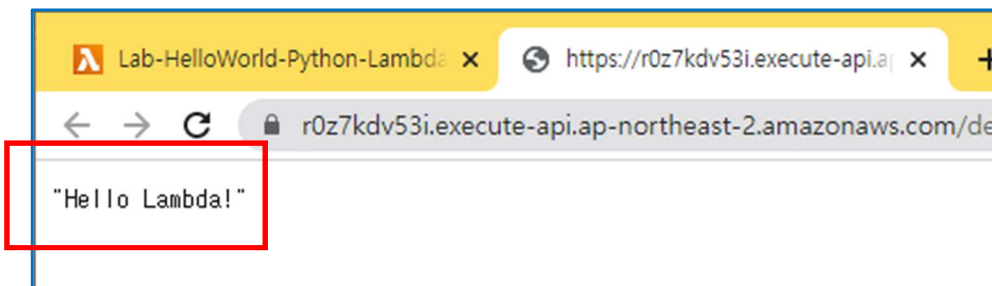
15. 다시 [코드]탭을 클릭하여 다음과 같이 코드를 수정했다. 그리고 [Deploy] 버튼을 클릭한다.



16. 수정한 코드의 업데이트가 끝나면, 다시 결과 화면에서 F5를 클릭하여 화면을 재시작한다.



17. 변경된 문자열이 rendering 되는 것을 확인할 수 있다.



18. 이번에는 새 함수를 생성하겠다. [함수 이름]은 "Lab-HelloWorld-Node-Lambda"로, [런타임]은 Node.js 16.x로 설정한 후, [함수 생성]을 클릭한다.

기본 정보

함수 이름

함수의 용도를 설명하는 이름을 입력합니다.

Lab-HelloWorld-Node-Lambda

공백 없이 문자, 숫자, 하이픈 또는 밑줄만 사용합니다.

런타임 정보

함수를 작성하는 데 사용할 언어를 선택합니다. 콘솔 코드 편집기는 Node.js, Python 및 Ruby만 지원합니다.

Node.js 16.x

권한 정보

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

▶ 고급 설정

취소

함수 생성

19. 새로운 함수가 잘 생성되었다. 이제 [트리거 추가]를 클릭한다.

함수 Lab-HelloWorld-Node-Lambda이(가) 생성되었습니다. 이제 함수의 코드 및 구성을 변경할 수 있습니다. 테스트 이벤트를 사용하여 함수를 호출하려면 [테스트]를 선택하십시오.

Lambda > 함수 > Lab-HelloWorld-Node-Lambda

Lab-HelloWorld-Node-Lambda

조절 ARN 복사 작업

▼ 함수 개요 정보

Lab-HelloWorld-Node-Lambda

Layers (0)

+ 트리거 추가

+ 대상 추가

설명

-

마지막 수정

19초 전

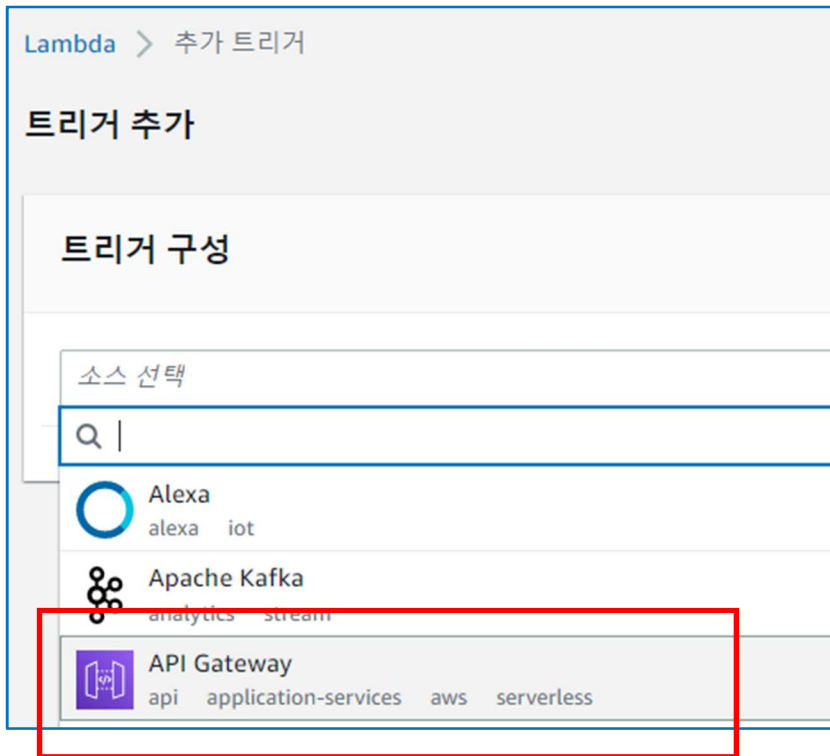
함수 ARN

arn:aws:lambda:ap-northeast-2:789534828835:function:Lab-HelloWorld-Node-Lambda

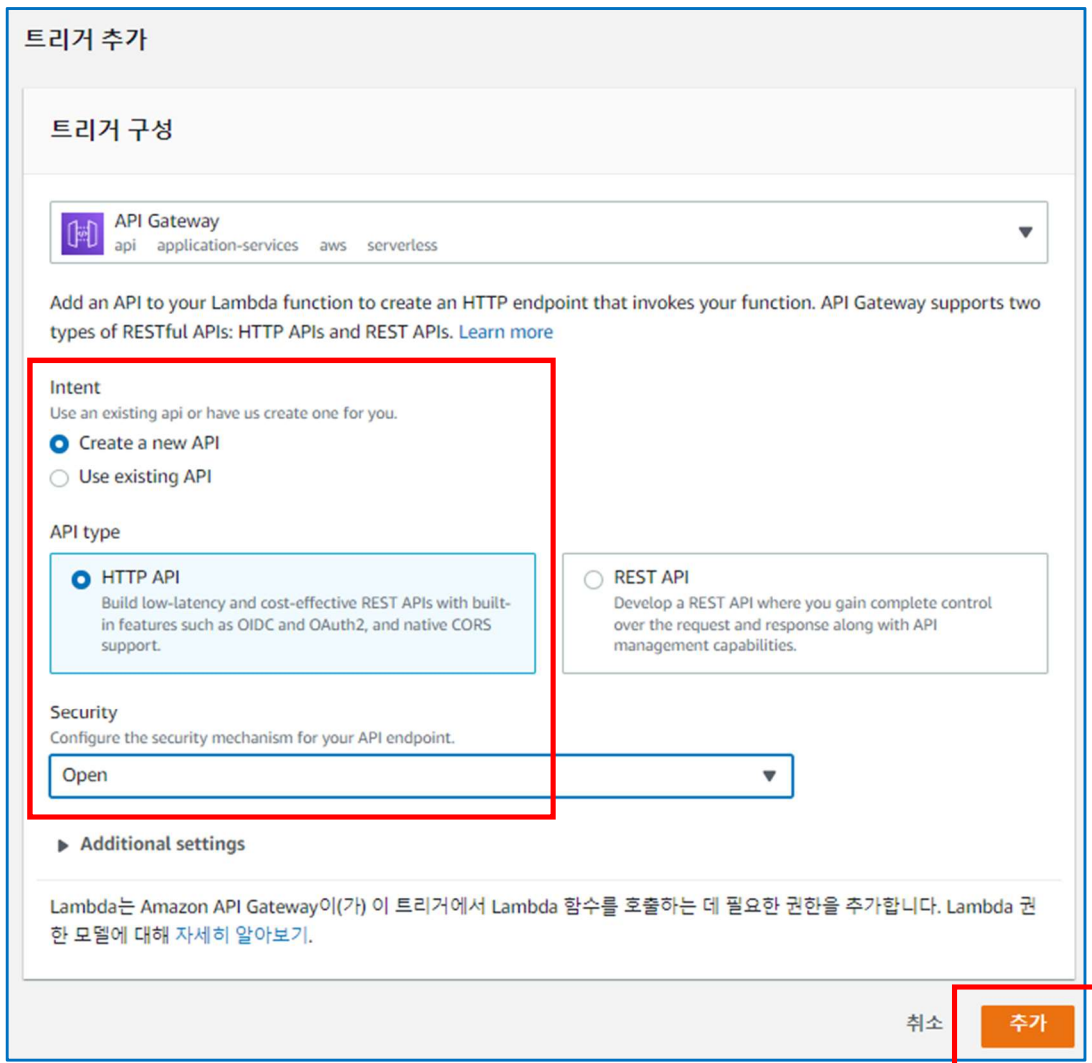
함수 URL 정보

-

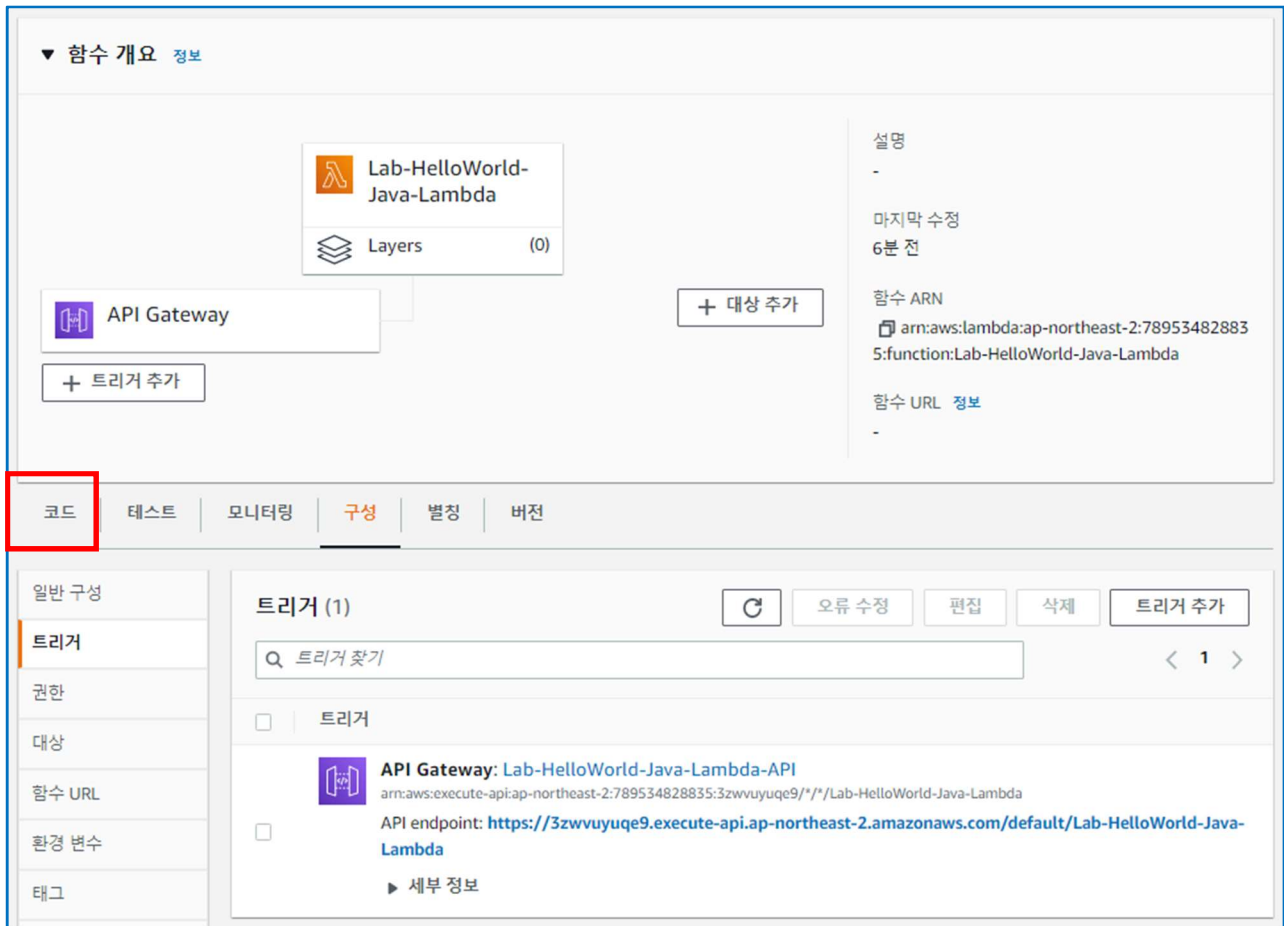
20. [트리거 추가]페이지에서, [트리거 구성] 섹션에서 [소스 선택]을 [API Gateway]로 선택했다.



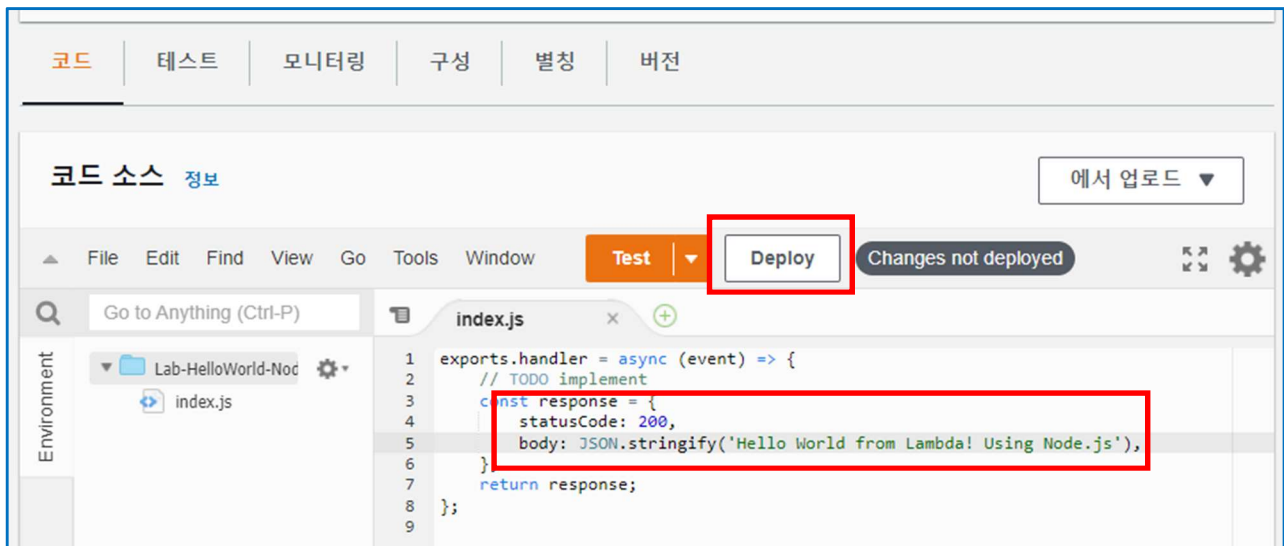
21. 이전에 생성했던 트리거와 같이 기본값을 설정하고 [추가] 버튼을 클릭한다.



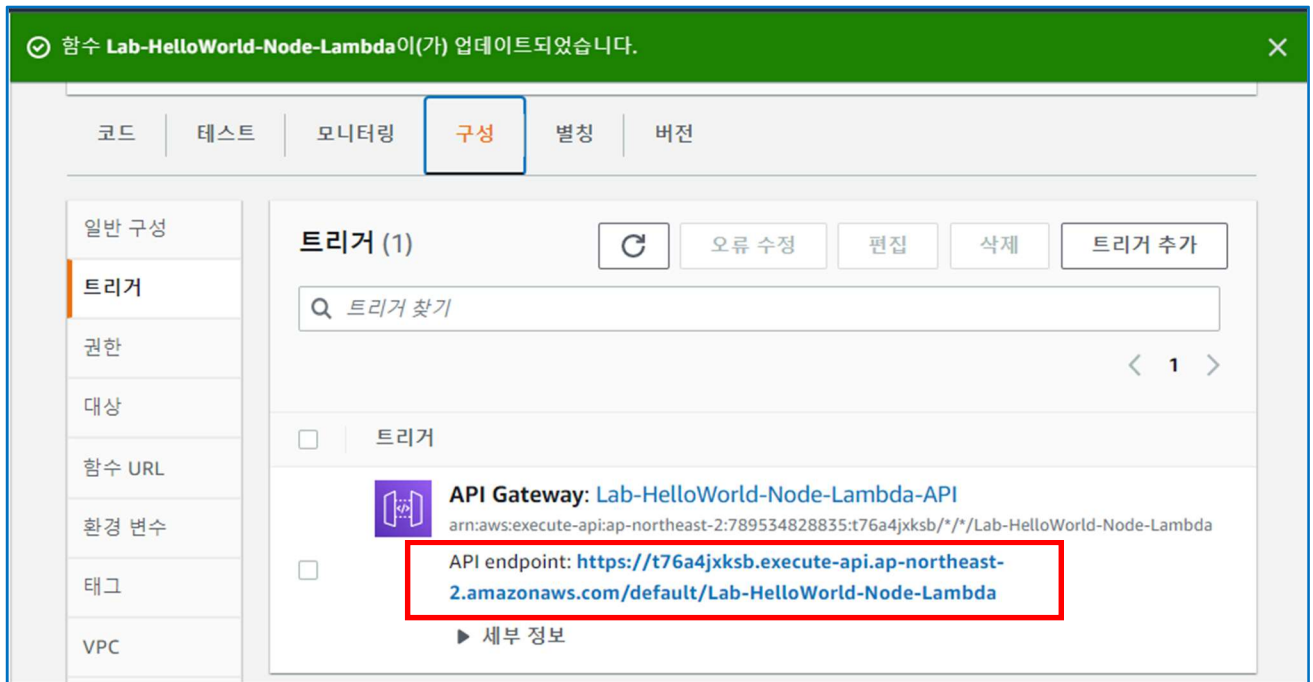
22. 트리거가 성공적으로 추가되었으면 [코드]을 클릭하여 함수의 소스를 확인해 보자.



23. 코드를 다음과 같이 수정하고 [Deploy] 버튼을 클릭한다.



24. 함수 코드의 수정이 끝나면 [API endpoint]의 링크를 클릭한다.



25. 우리는 불과 수 분만에 간단한 코드 편집만으로 웹 서버를 생성할 수 있었다.

