STT 서비스 구축을 위한 AWS 설정 과정

학과: 정보컴퓨터공학부

과목명: 클라우드컴퓨팅

7조 김태훈 박민재 김성문 박재열

1. 목적

   STT 서비스 구축을 위해 AWS에서 제공하는 서비스를 생성 및 설정하며, 서비스들끼리 연결시킨다.

2. 전체 다이어그램

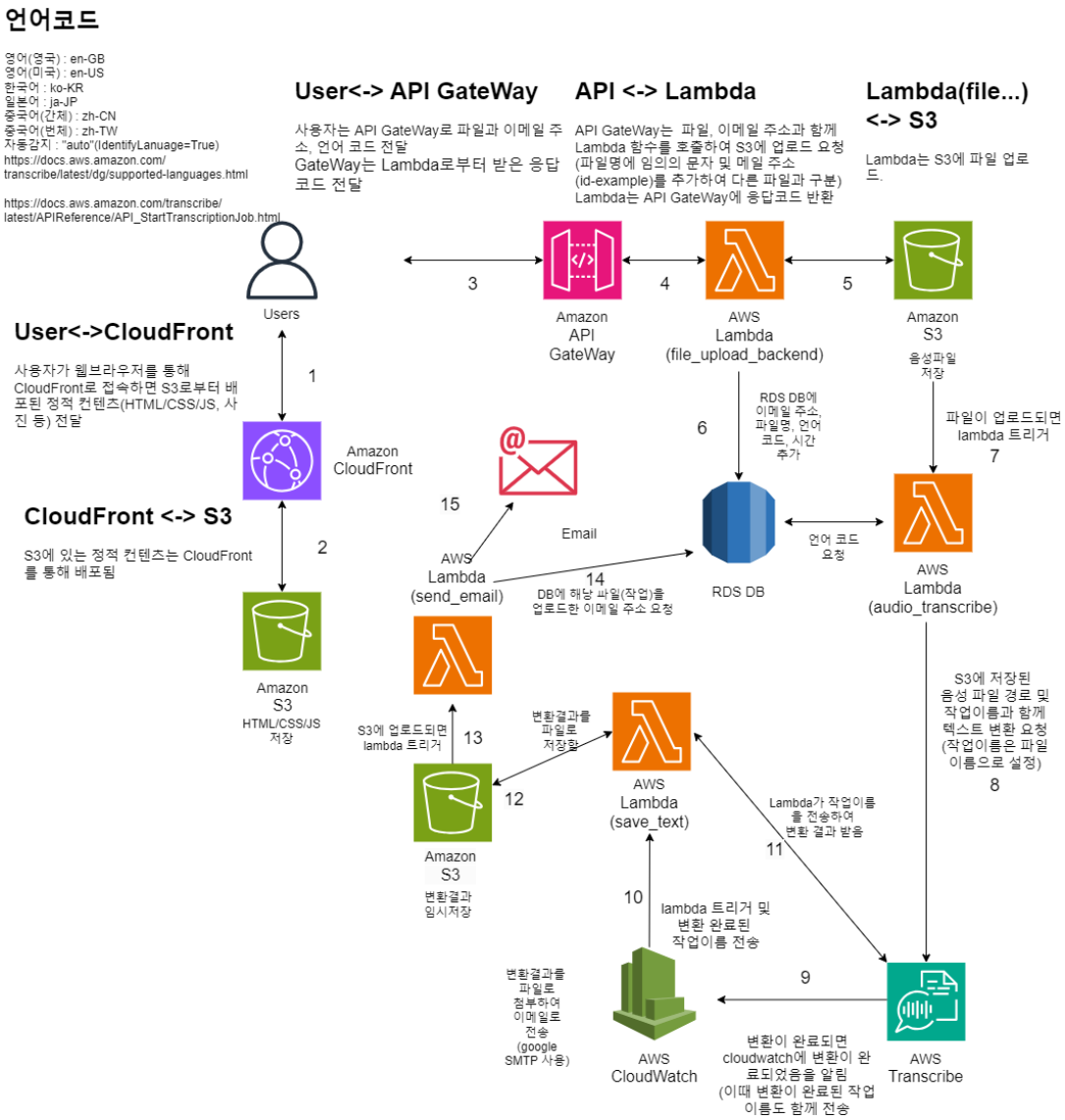


그림 1 전체 다이어그램

   전체 다이어그램은 위와 같다.

3. 정적 웹사이트 구축

   Amazon S3, Amazon CloudFront를 사용하여 정적인 파일(html/css/js)를 호스팅하는 웹페이지를 구축한다.

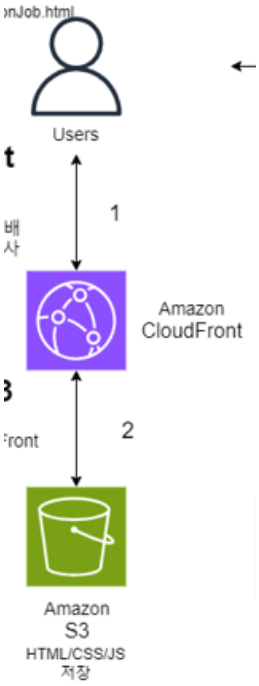
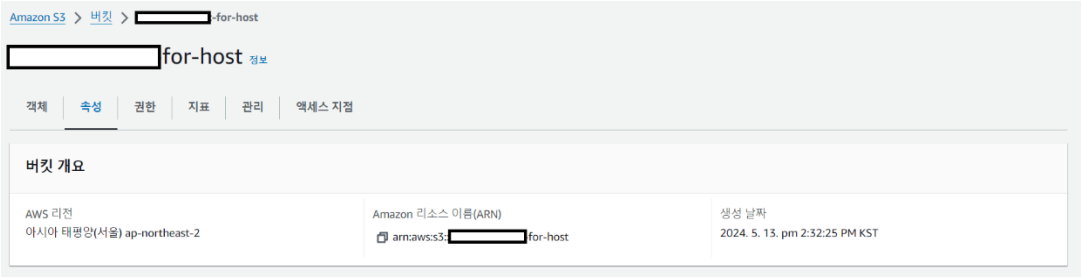


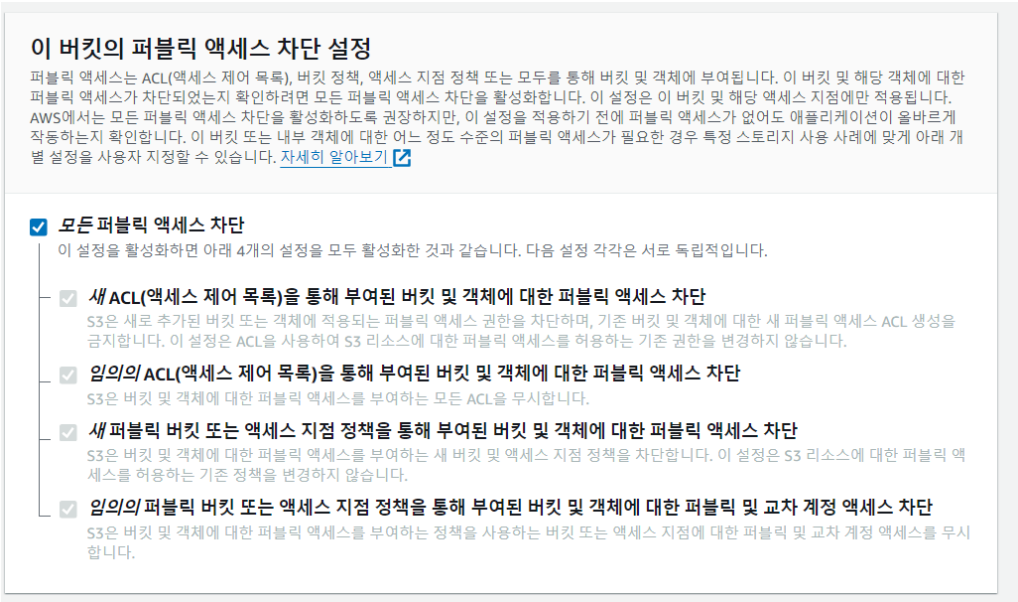
그림 2 정적 웹사이트 구축

즉 그림1에서 위 부분을 구성한다.

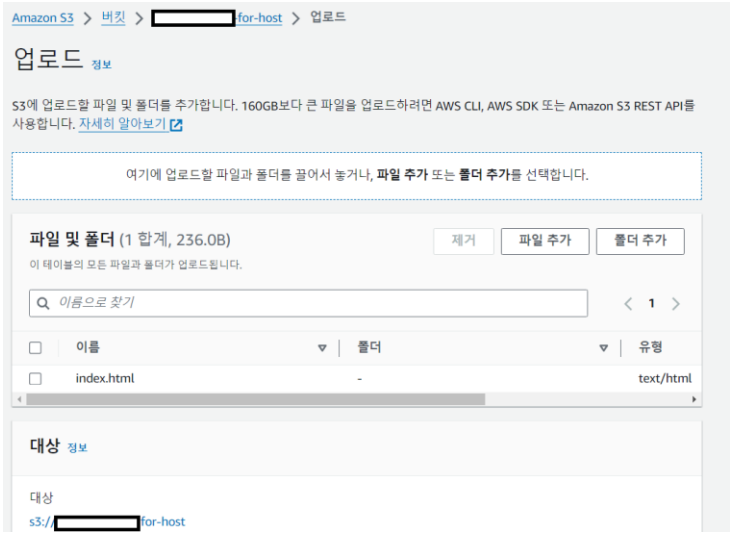
(1) s3 버킷 생성



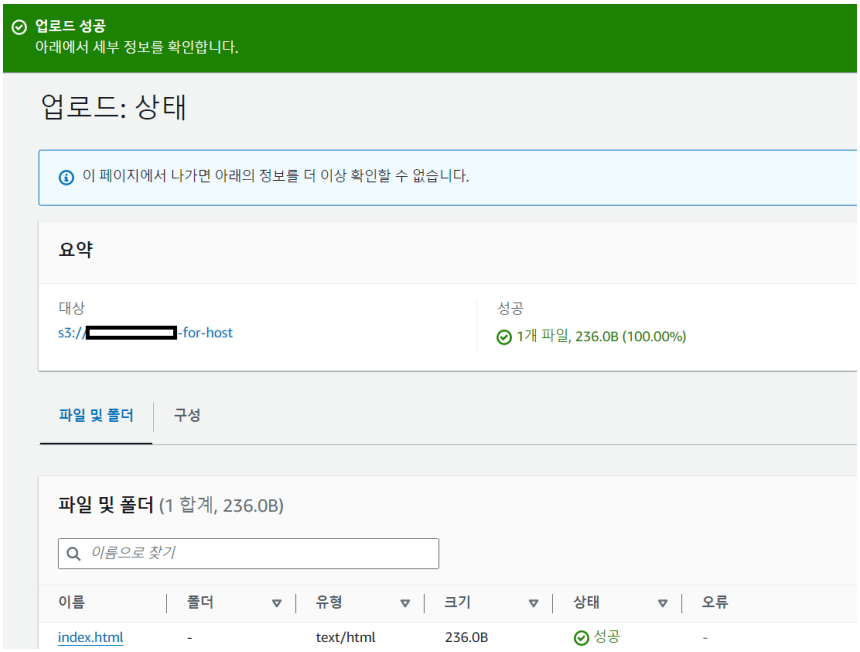
버킷 이름을 지정하여 버킷을 만든다.



cloudfront를 이용하여 서비스할 예정이므로, 퍼블릭엑세스는 차단한다.



s3에 index.html 파일을 업로드한다(테스트용).



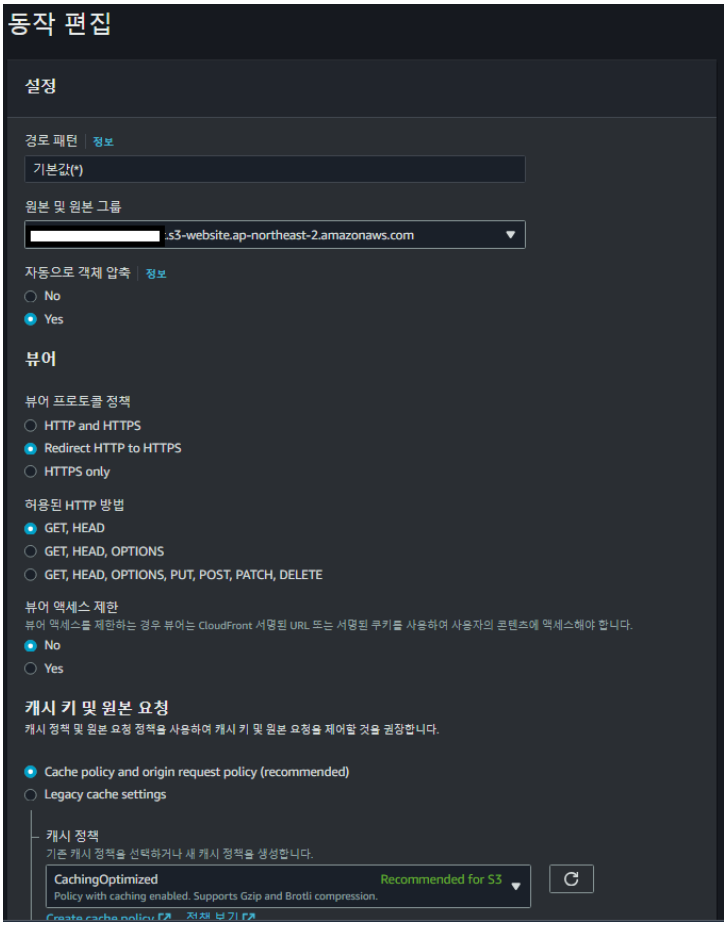
업로드 성공 메시지를 확인한다.

s3의 속성에서 정적 웹사이트 호스팅 설정을 편집한다.

위와 같이 정적 웹사이트 호스팅을 활성화하고, index.html을 기본 페이지로 설정한다.

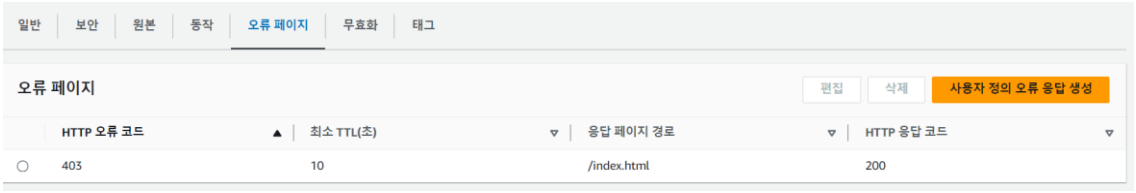
(2) cloudfront 설정

그리고 cloudfront를 설정한다. origin domain은 정적 웹사이트 호스팅을 활성화한 s3로 설정하고, OAC를 설정한다(후술). 그리고 Origin Shield를 활성화한다.



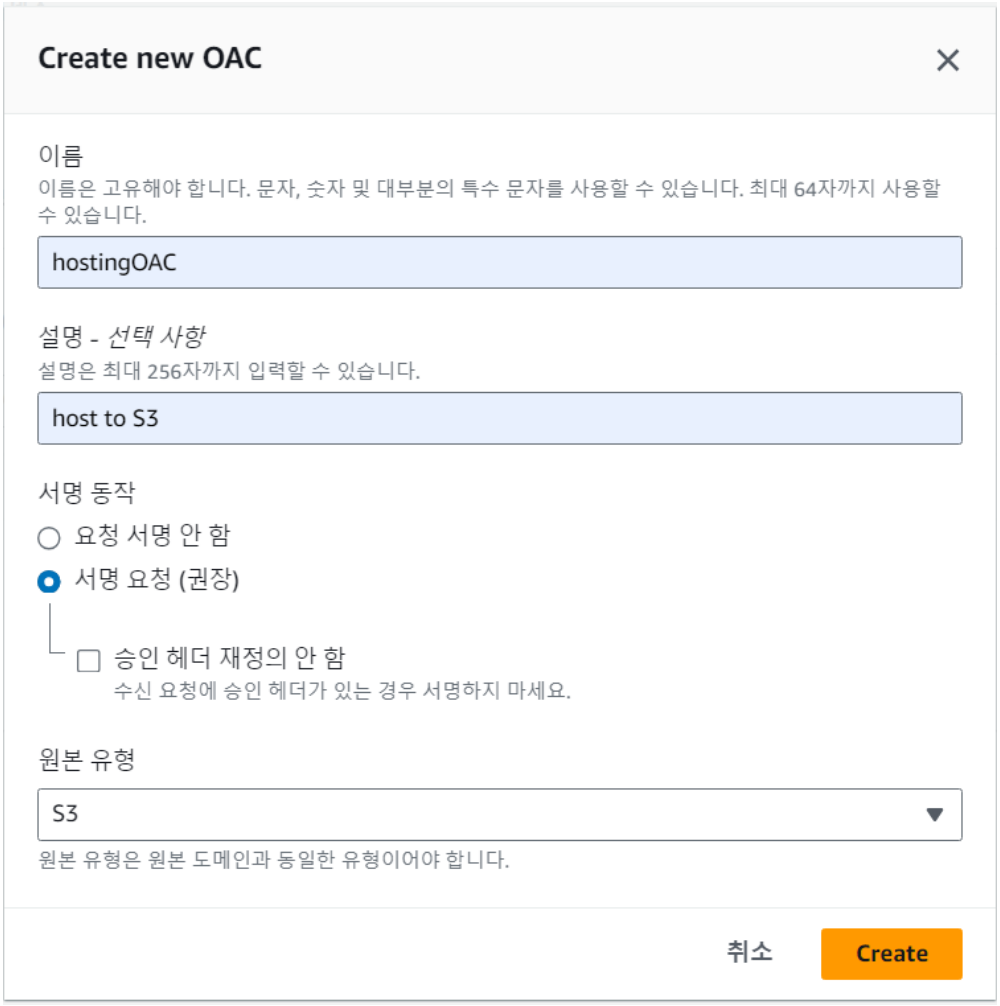
그리고 cloudfront의 동작을 편집한다. GET, HEAD를 허용하고, 캐시 정책을 CachingOptimized로 설정하여 cloudfront가 s3의 컨텐츠를 캐싱하도록 한다.

모든 엣지 로케이션에 컨텐츠를 배포하고, xxxx.cloudfront.net으로 접속 시 index.html을 반환하도록 한다. HTTP/2 를 지원하도록 설정하고, IPv6를 켠다.

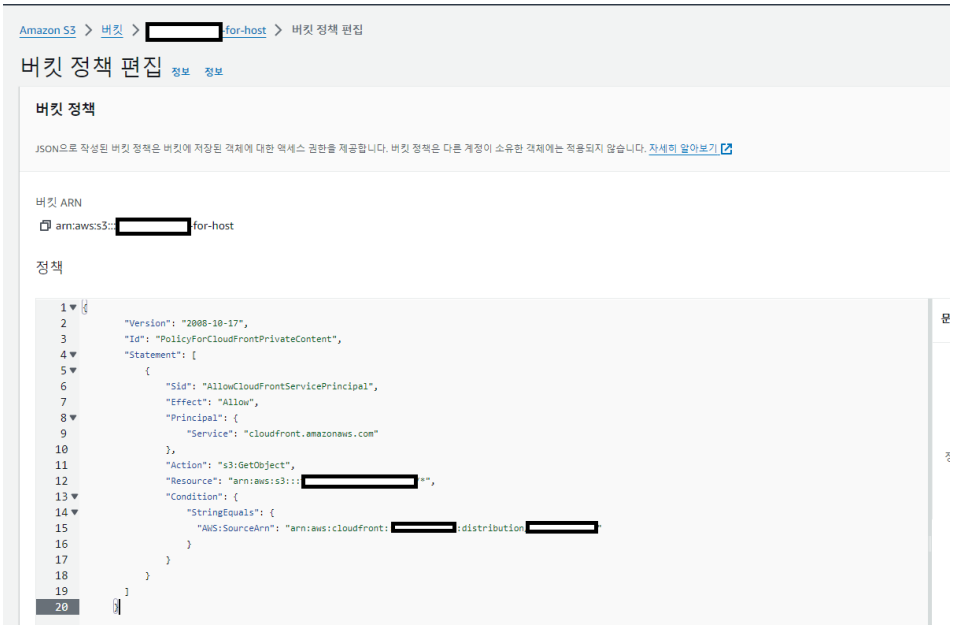


사용자가 정의되지 않은 리소스를 호출했을 경우, s3에서 403 forbidden을 반환하게 되고, cloudfront에서는 이를 index.html(응답 코드 200)로 사용자에게 응답한다. 즉 정의되지 않은 리소스를 호출하면 사용자는 초기화면(index.html)로 돌아간다.

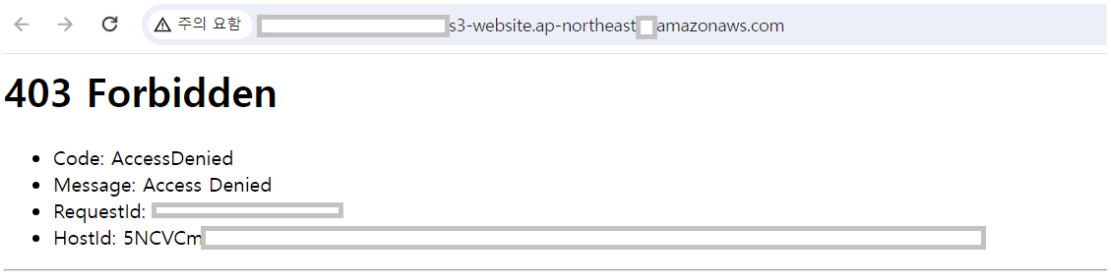
(3) OAC 생성 및 버킷 정책 편집



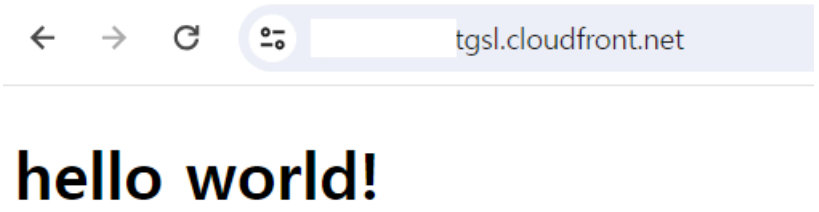
OAC를 생성한다 OAC를 생성하여 cloudfront가 s3에 접근할 수 있도록 한다.



OAC를 생성하면 s3 버킷 정책을 편집하라고 알려준다. s3 버킷 정책에 넣을 내용과 함께 안내되므로, 이를 s3 버킷 정책에 입력한다.

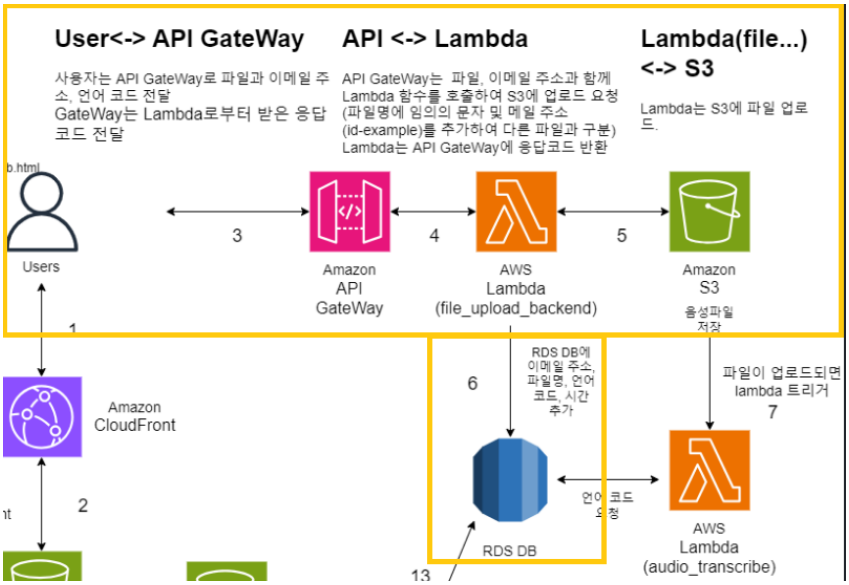


s3로 바로 접속하면 403 Forbidden 메시지와 함께 접속이 차단되며, https도 지원되지 않는다.



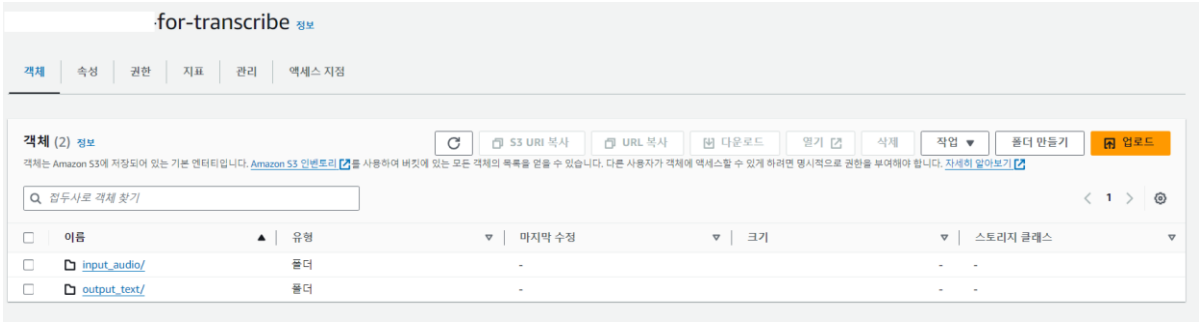
cloudfront로 접속하면 위와 같이 잘 접속되는 것을 확인할 수 있고, https도 지원되는 것을 확인하수 있다.

4. 파일 업로드 구축



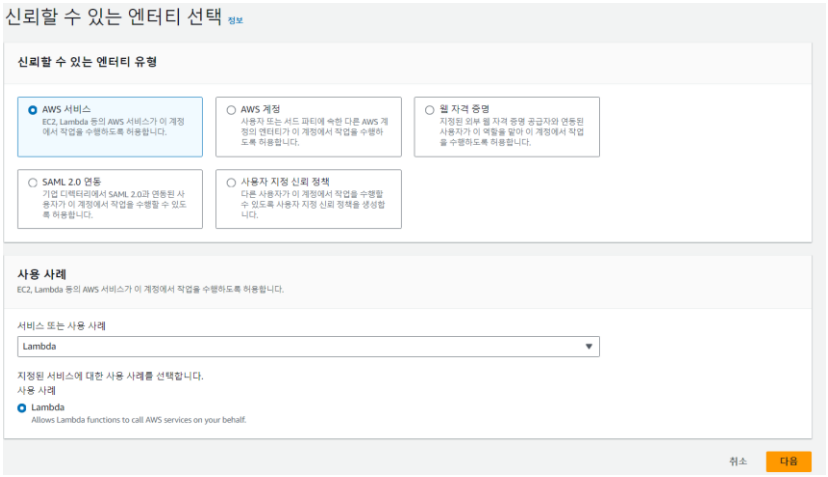
이제 위와 같이 다이어그램에서 API GateWay, file_backend_lambda 함수, 음성파일 저장하는 S3, 음성파일 관련정보(이메일, 파일명, 언어코드, 업로드 시간)를 저장하는 RDS DB(mysql)를 만든다.

(1) 버킷 생성

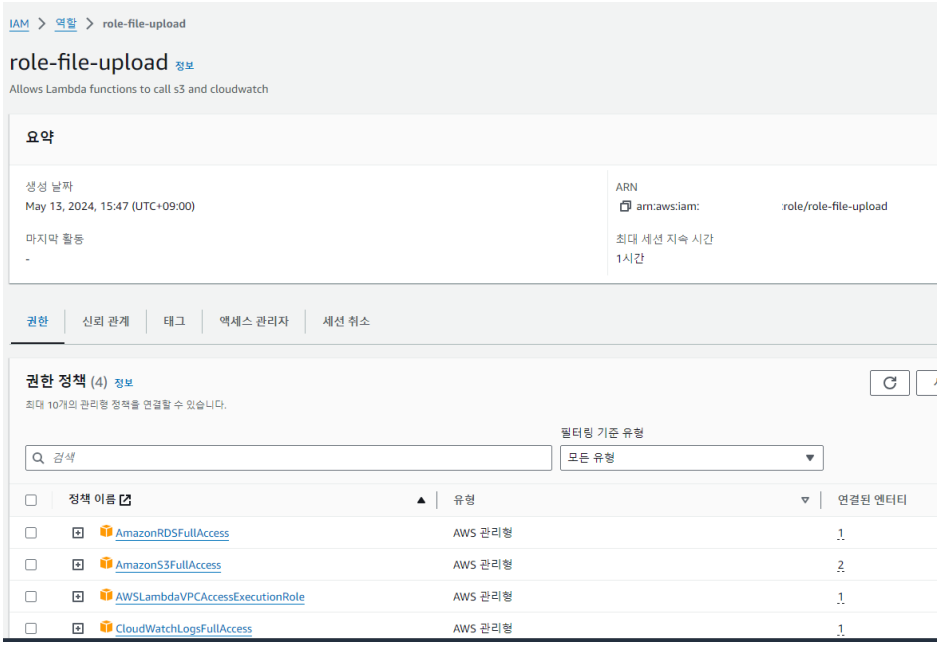


s3 버킷을 만들고, input_audio와 output_text 폴더를 만든다. input_audio는 업로드된 음성 파일이, output_text는 변환 결과가 임시 저장되는 곳이다. 속성 탭에서, 모든 퍼블릭 엑세스를 차단해야한다.

(2) lambda 함수에 적용할 IAM 생성



file_upload_backend 람다 함수를 위한 IAM를 만든다.



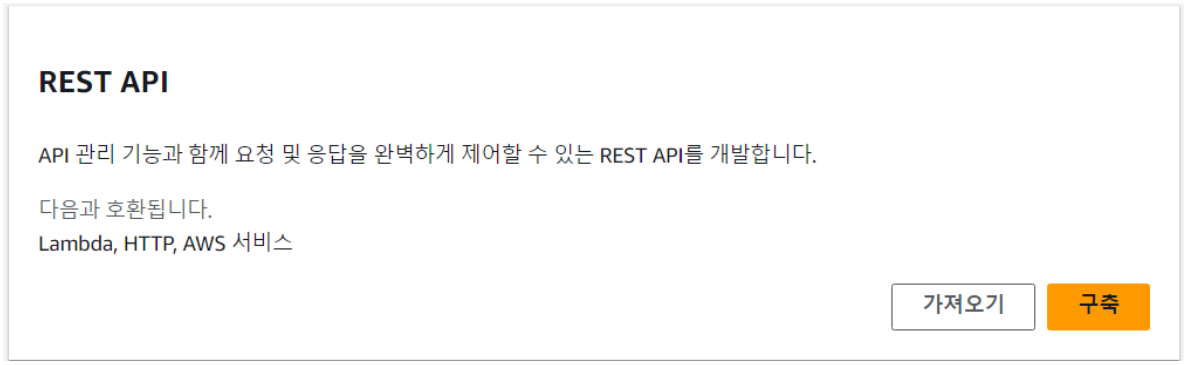
file_upload_backend 람다 함수를 위한 IAM은 RDS와 S3에 접근하여야 되고, RDS DB가 속한 VPC에 접속할수 있어야 한다.

(3) lambda 함수 생성

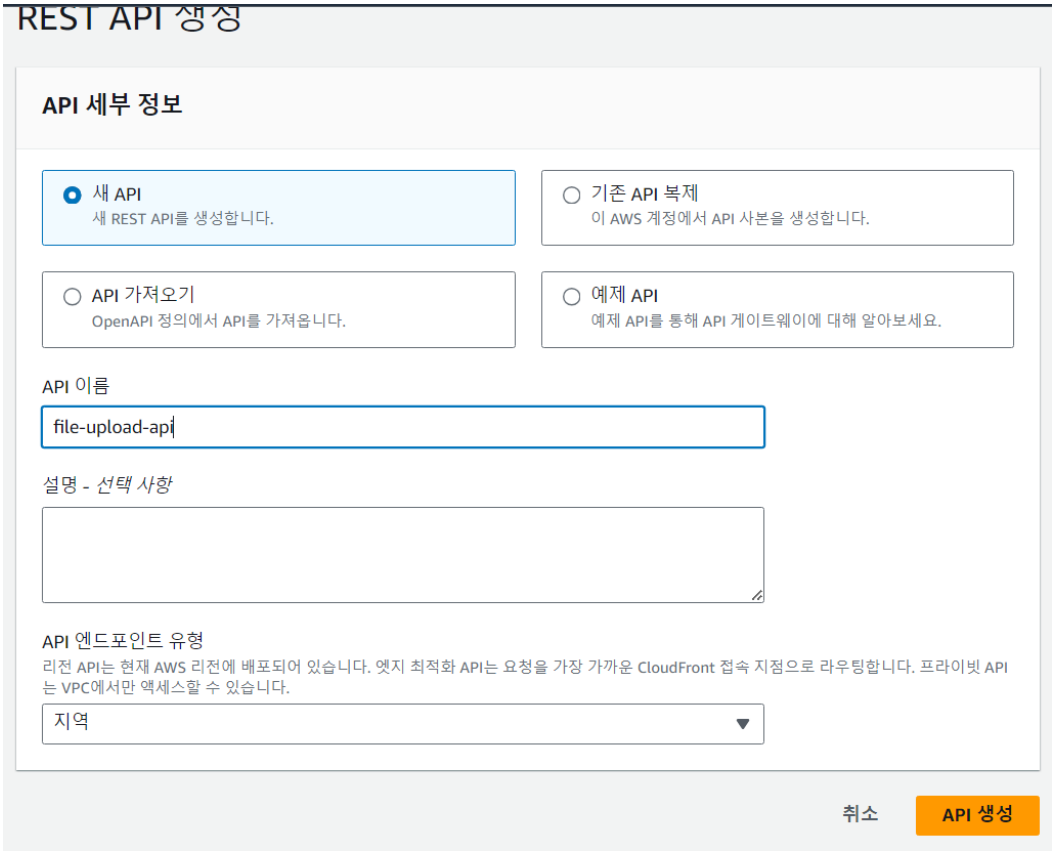
이제 함수를 생성한다. 런타임은 node.js 20 x로 설정하고, 역할을 아까 만든 role-file-upload로 설정한다.

lambda 함수를 만든 후. 설정에서 메모리와 제한시간을 변경한다. 메모리는 기존 128MB에서 512MB로, 제한 시간은 기존 3초에서 1분 30초로 변경한다. s3에 파일을 저장하고, RDS DB에 데이터를 써야하므로 , 제한 시간을 늘렸다.

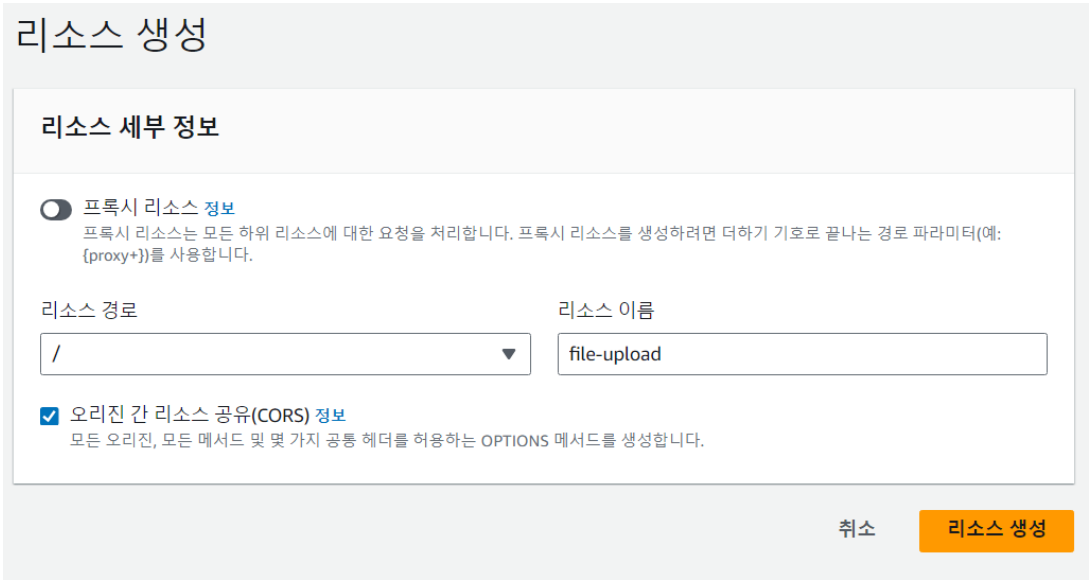
(4) API 생성



이제 API GateWay 메뉴로 가서, REST API를 구축한다.



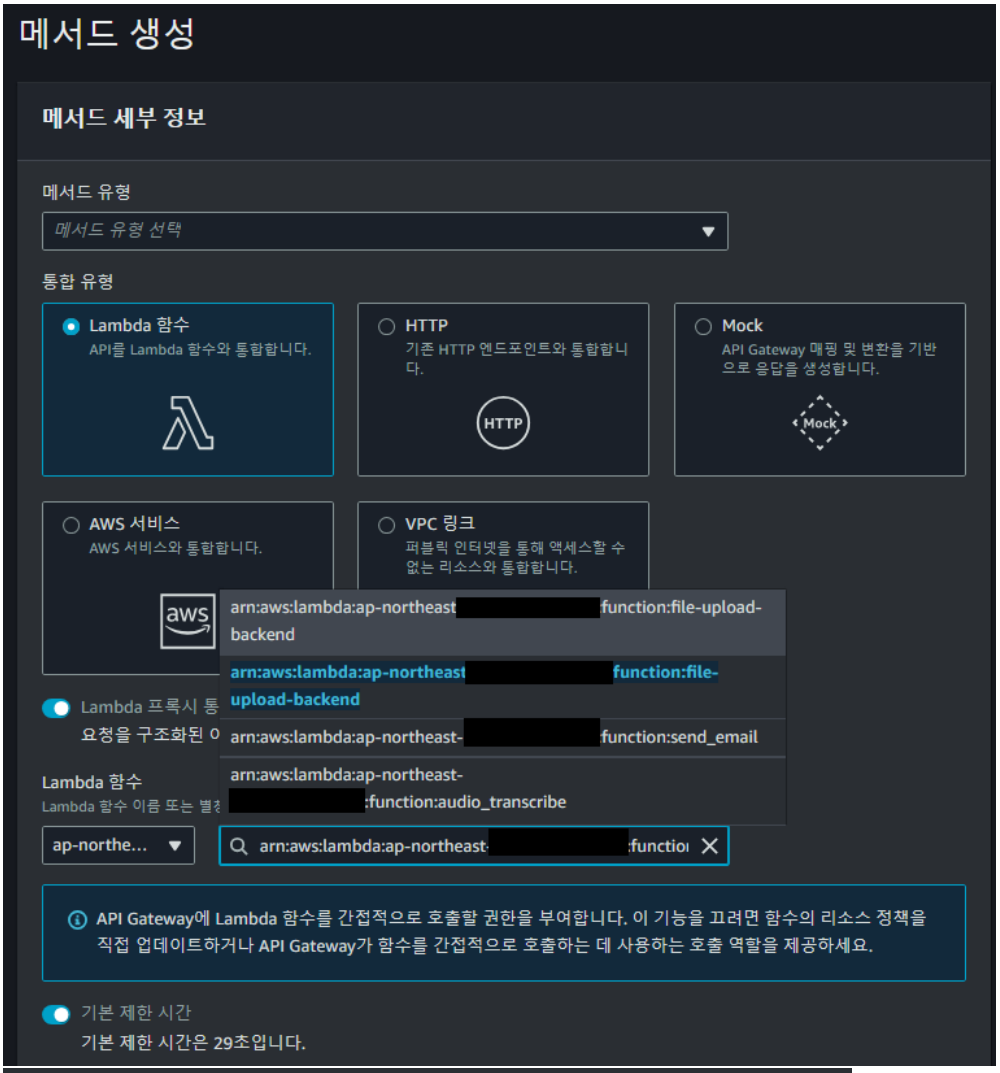
API 세부 정보에서 이름은 file-upload-api로 설정하고, API 엔드포인트 유형은 지역으로 설정한다.

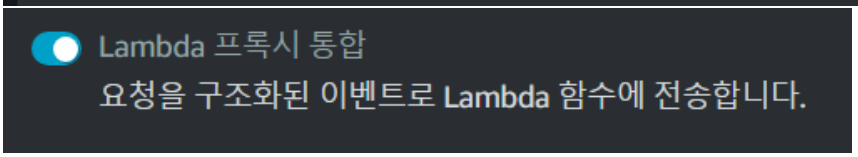


만들어진 API에 리소스 생성을 클릭하여 하위 리소스를 생성한다. 즉 호출될 때

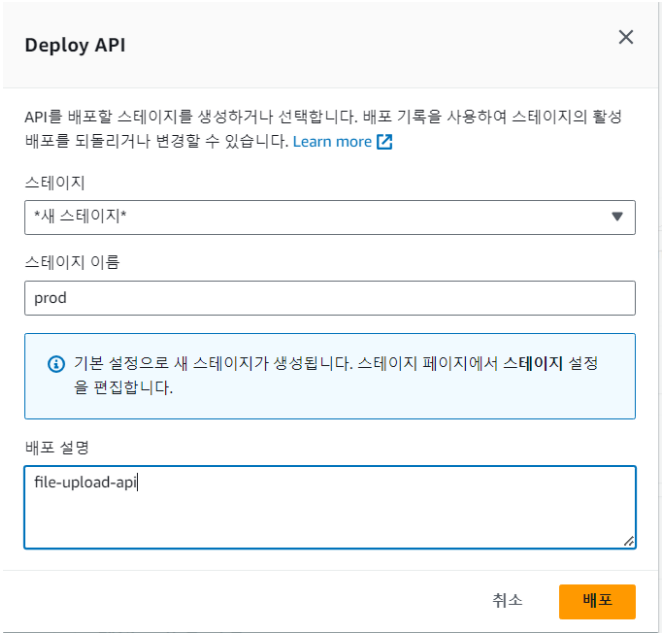
xxxxxx.execute-api.ap-xxxxxxxxx.amazonaws.com/(배포 스테이지)/file-upload로 호출된다. (배포 스테이지는 배포할 때 설정)

하위 리소스를 생성할 때 CORS를 체크한다.



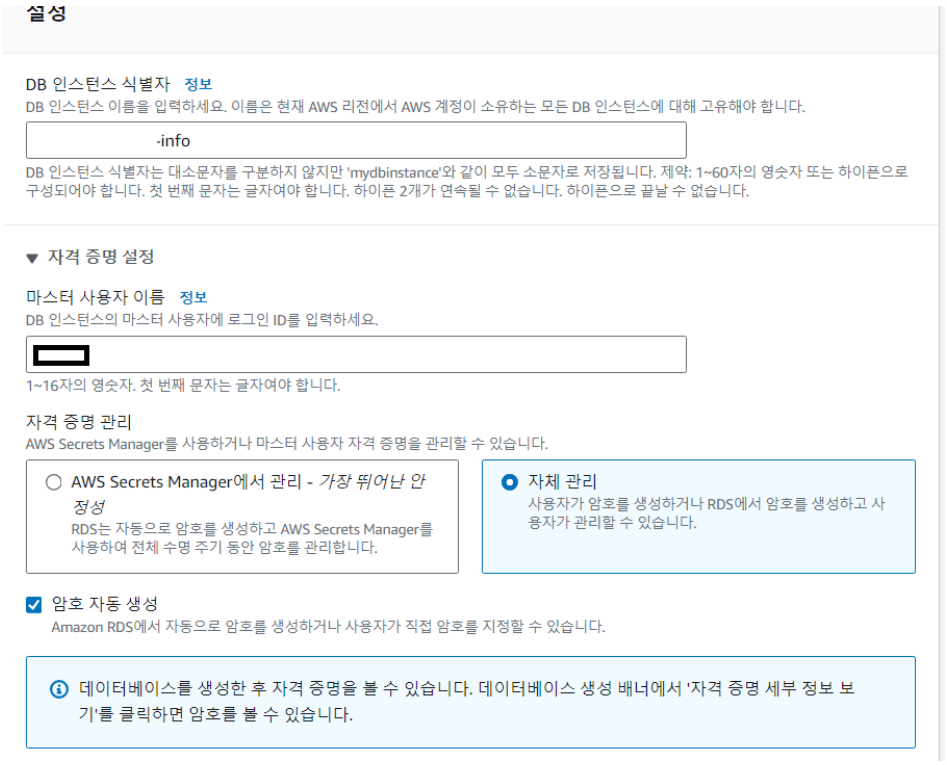


그리고 API가 실행할 것을 선택한다. 여기서는 lambda 함수를 실행하고, lambda 함수 중 file-upload-backend 함수를 실행하는 것으로 설정한다. 그리고, multipart/form-data를 받기 위해 lambda 프록시 통합을 활성화한다.

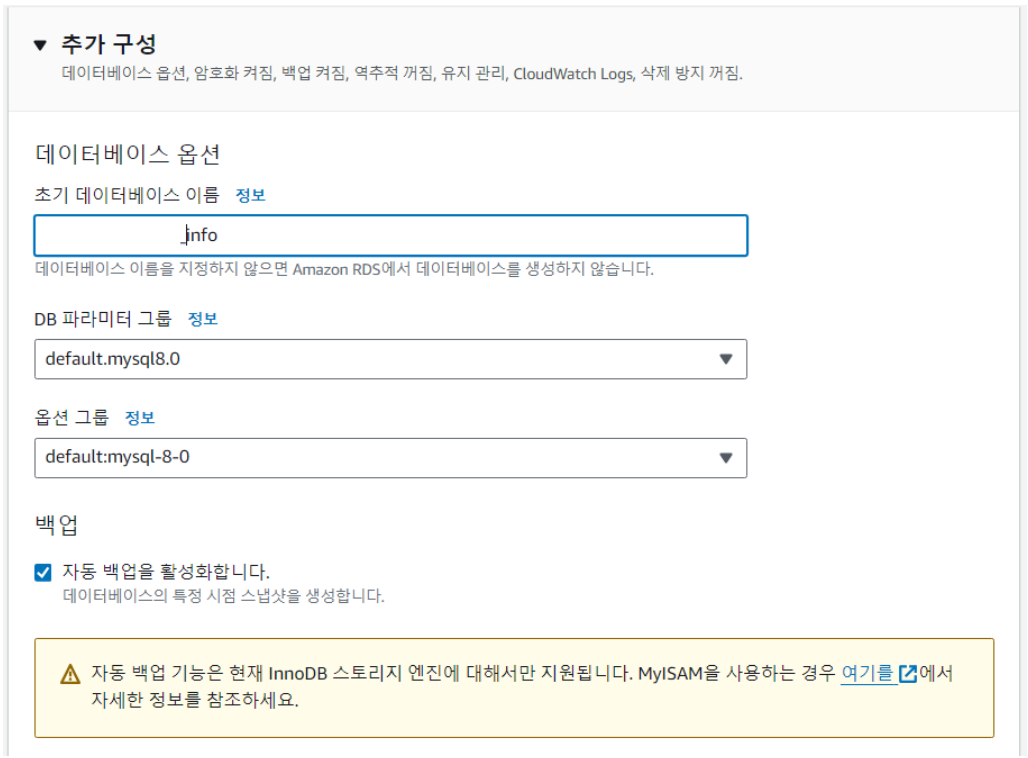


그리고 위와 같이 API를 배포한다. 스테이지 이름을 prod로 했으므로, 파일 업로드 요청시 API 주소는 xxxxxx.execute-api.ap-xxxxxxxxx.amazonaws.com/prod/file-upload 가 된다.

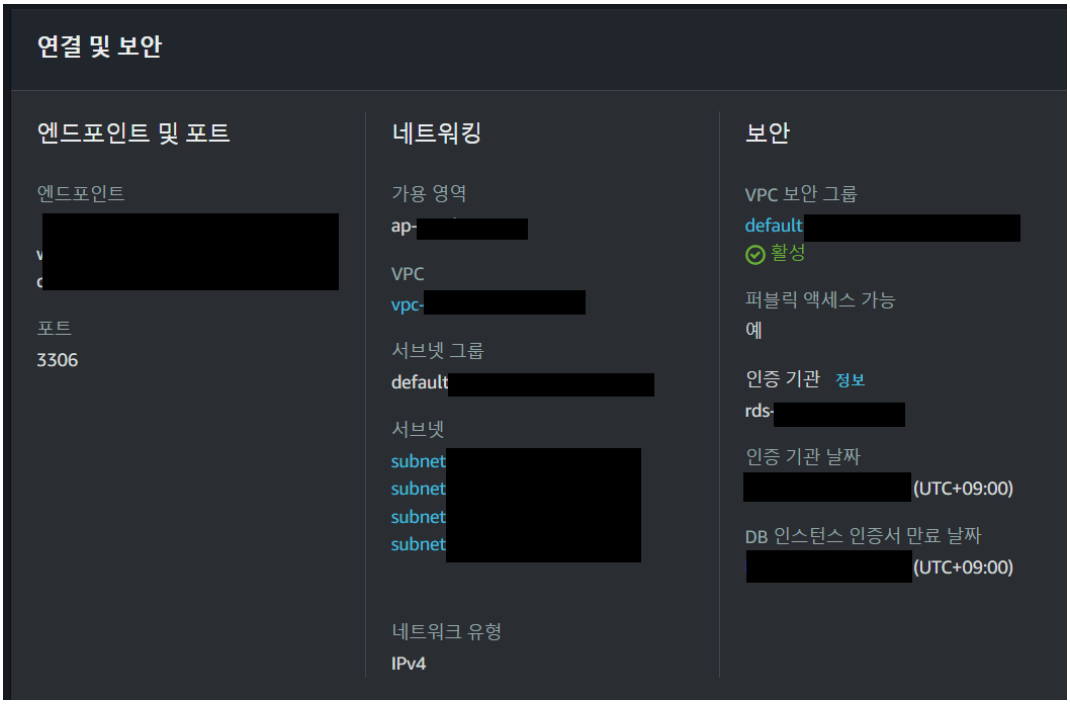
(5) RDS 생성



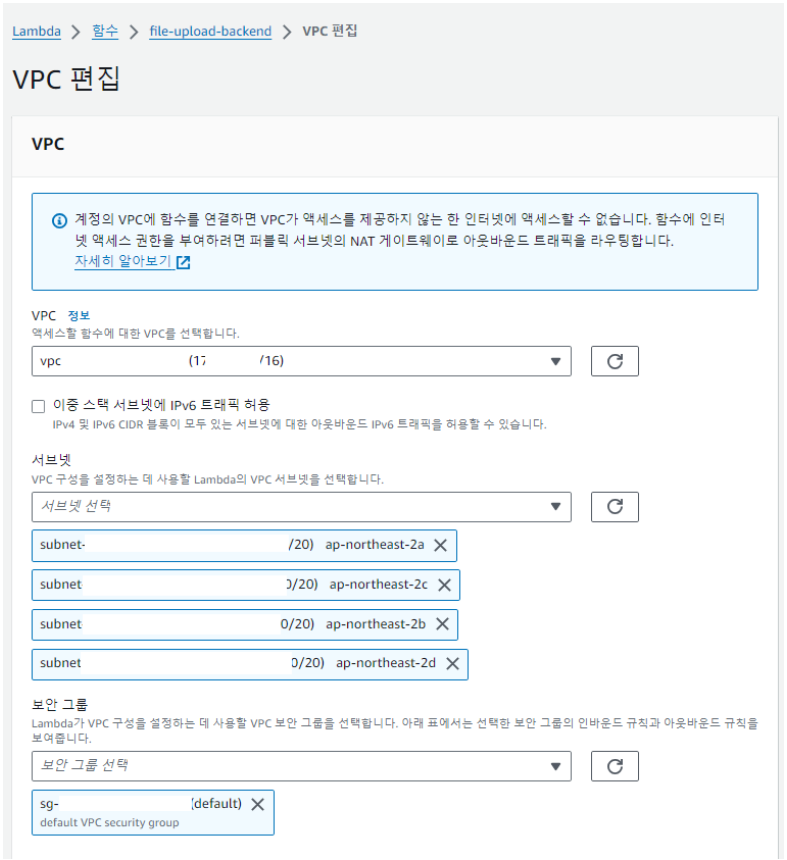
이제 DB를 생성한다. DB 인스턴스 식별자를 설정하고, 마스터 사용자 이름과 비밀번호를 설정한다. 암호 자동 생성을 체크하면 생성 후에, 비밀번호가 담겨진 파일을 다운받게 되고, 이 안에 비밀번호가 담겨있게 된다. 이 파일을 메모장으로 열면 무작위의 숫자와 문자로 이루어진 비밀번호가 들어있다.



DB는 mysql로 설정하고, 자동 백업을 활성화한다.



그리고 위와 같이 RDS가 속할 VPC와 VPC 보안 그룹을 설정한다. 이 설정을 통해 VPC에 속한 사용자/lambda 함수만 DB에 접속할 수 있게 한다.

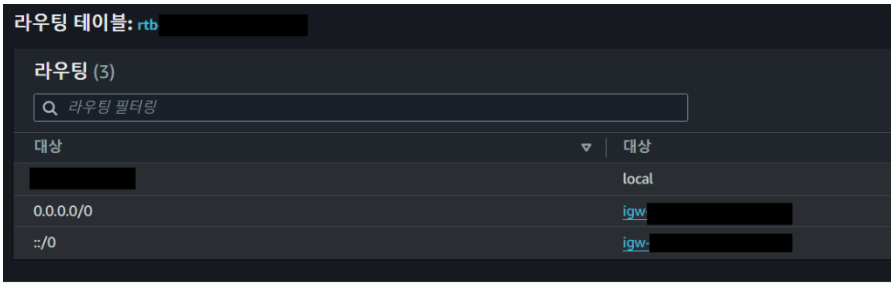


이전에 만들었던 lambda 함수에서 VPC 편집에 접속하여, RDS가 속한 VPC와 보안 그룹으로 설정한다.

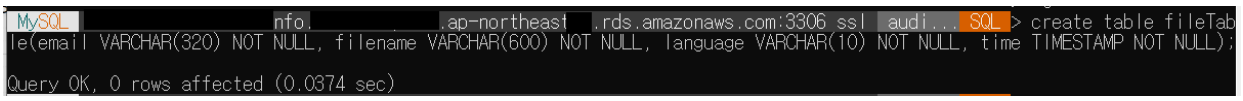
VPC 보안 그룹의 인바운드 규칙은 위와 같다. 보안 그룹에 속한 모든 트래픽 및 mysql로 접속하는 나의 IP는 허용하도록 설정하였다.

https://hossamelshahawi.com/2021/10/24/fix-errno97-address-family-not-supported-by-protocol/

를 참고하여 vpc에 ipv6 주소를 추가한다.

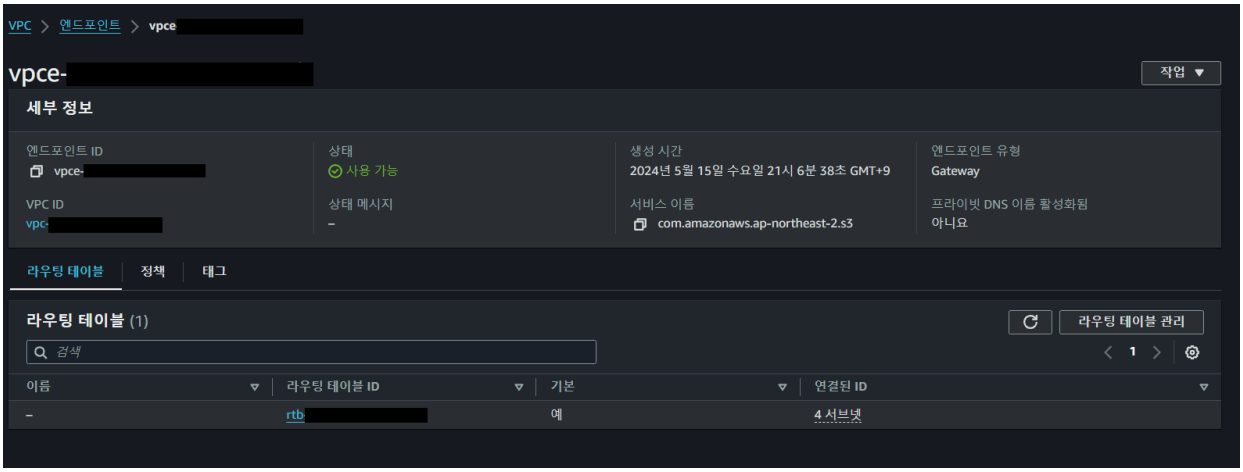


그리고 vpc 라우팅테이블에 ipv6를 인터넷 게이트웨이에 연결한다.

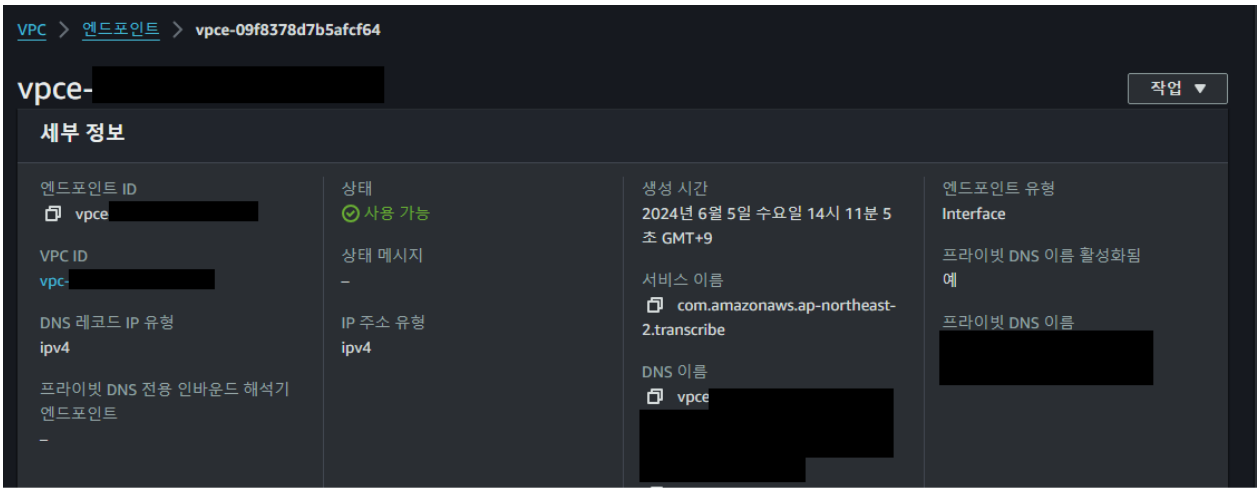


mysql shell을 이용하여 접속하여 database와 테이블을 생성한다. 테이블은 위와 같이 생성하며, 구체적으로는 아래와 같다,

fileTable(email VARCHAR(32) NOT NULL, filename VARCHAR(600) NOT NULL, language VARCHAR(10) NOT NULL, time TIMESTAMP NOT NULL)

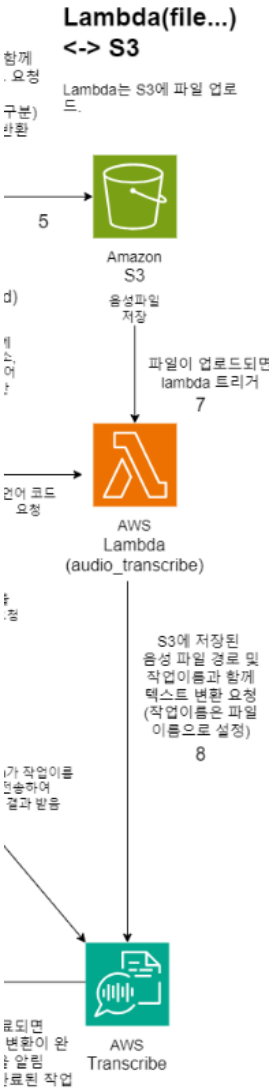


또한 vpc 엔드포인트를 생성하여, vpc에 속한 lambda 함수가 S3에 접속할 수 있도록 한다.



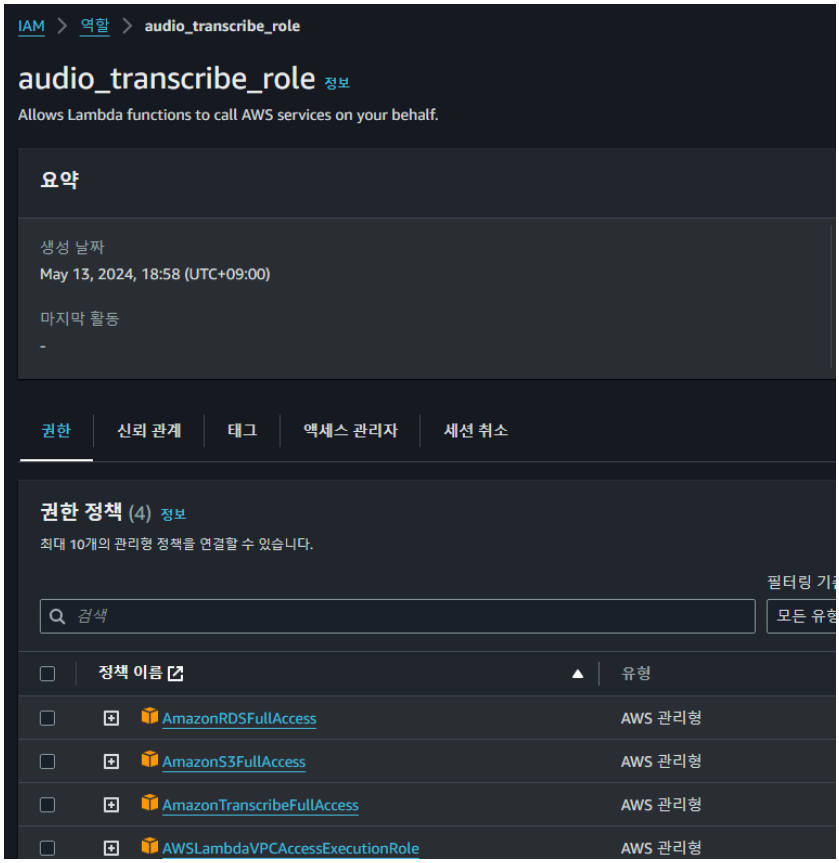
vpc 엔드포인트를 생성하여, vpc에 속한 lambda 함수가 transcribe에 접속할 수 있도록 한다.

5. transcribe 구축



이제 다이어그램에서 S3가 audio_transcribe 람다 함수를 트리거하는 것, 해당 lambda 함수가 AWS Transcribe에 변환 요청하는 것을 구축한다.

(1) IAM 생성

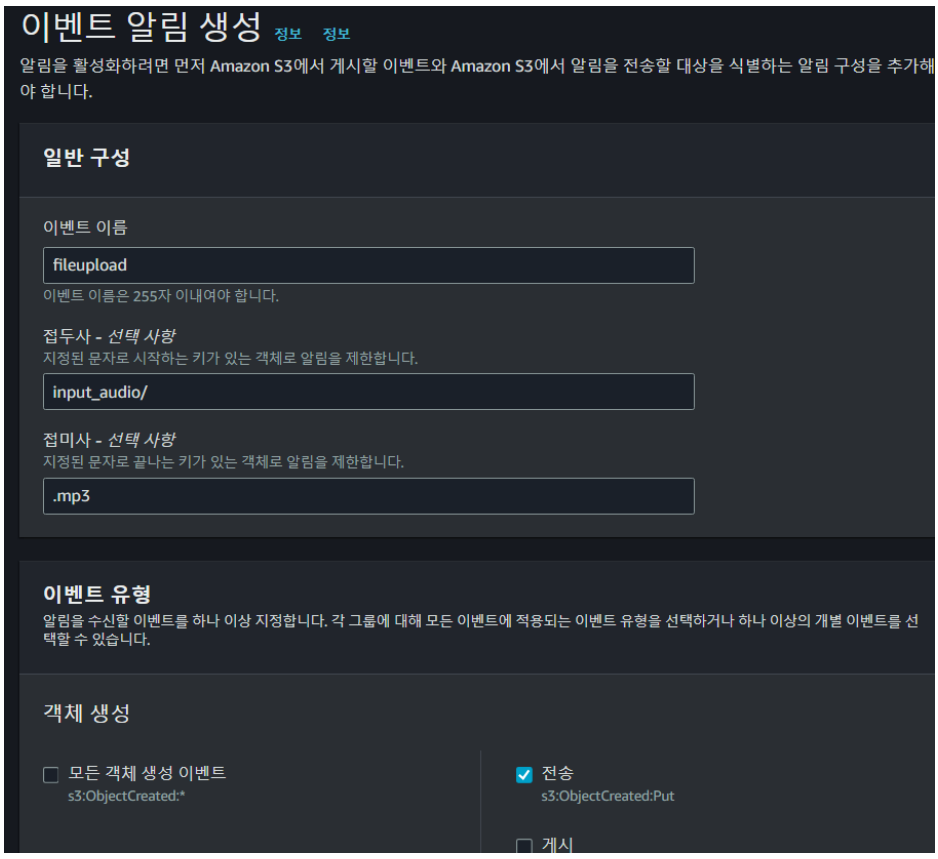


audio_transcribe에 적용할 IAM을 생성한다. 이 람다 함수는 RDS, S3, Transcribe에 FullAccess가 가능해야 하고, VPC에 접근할 수 있도록 한다.

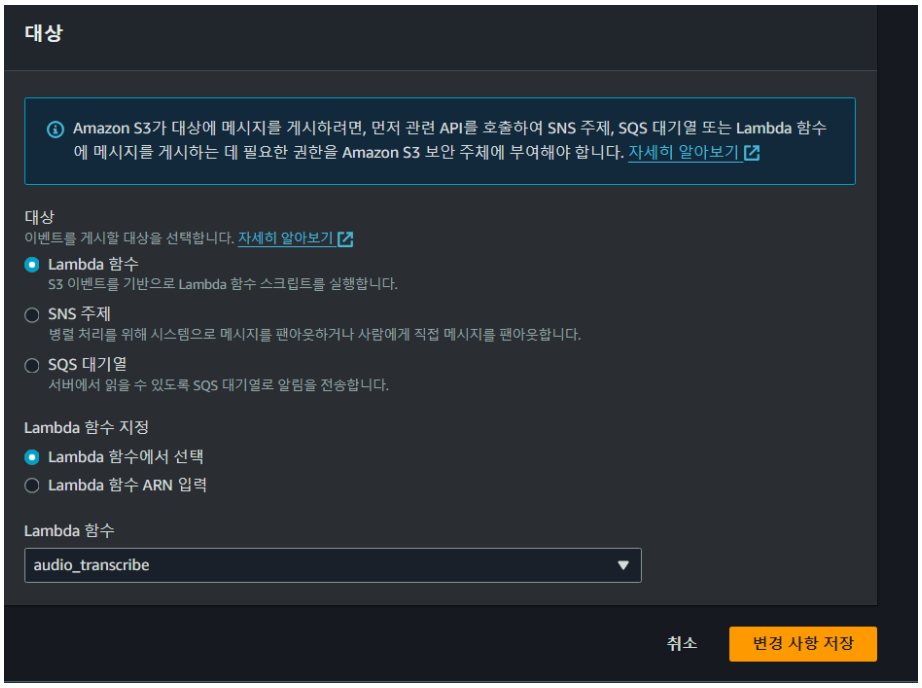
(2) lambda 함수 생성

위와 같이 audio_transcribe 람다 함수를 python 3.12로 생성한다.

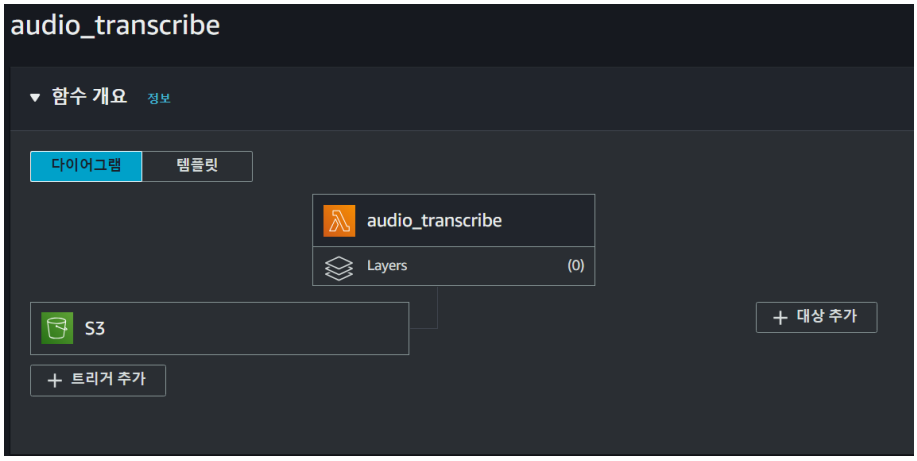
(3) 이벤트 알림 생성



s3의 속성에서 이벤트 알림을 생성한다. s3에 input_audio 폴더에 mp3 파일이 put되면 이벤트가 발생하도록 설정한다.

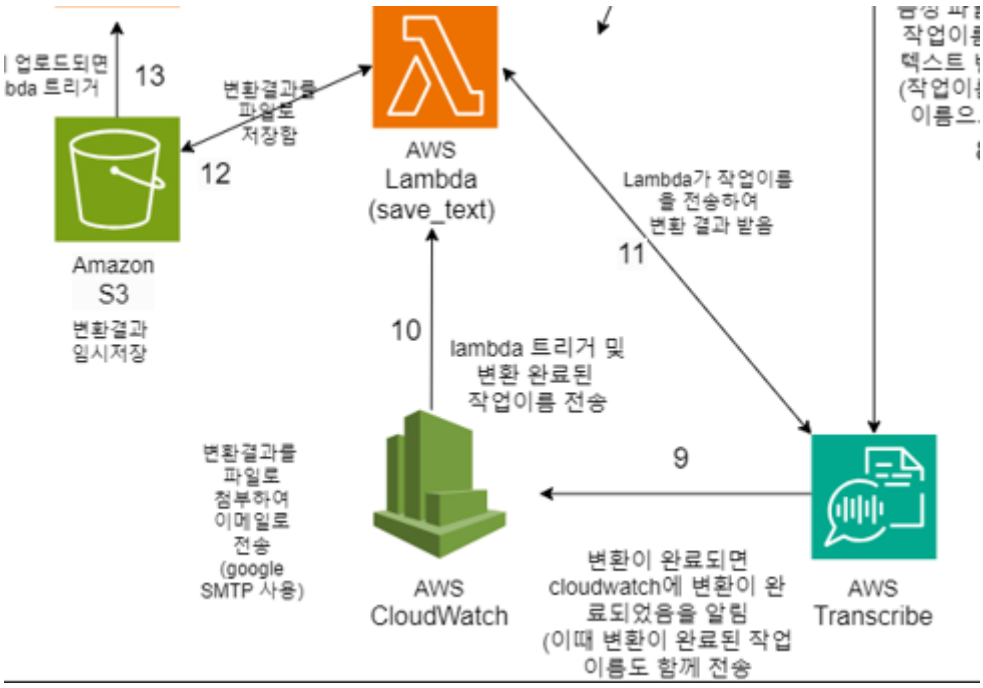


그리고 해당 이벤트는 audio_transcribe를 호출하도록 설정한다.



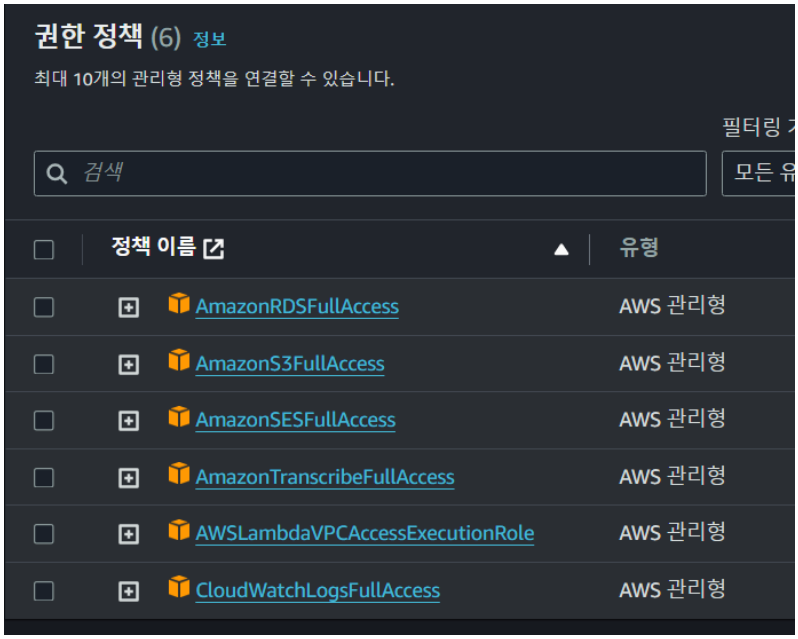
이제 람다 함수에서 위와 같이 S3가 트리거로 설정된 것을 확인할 수 있다.

6. save_text 구축

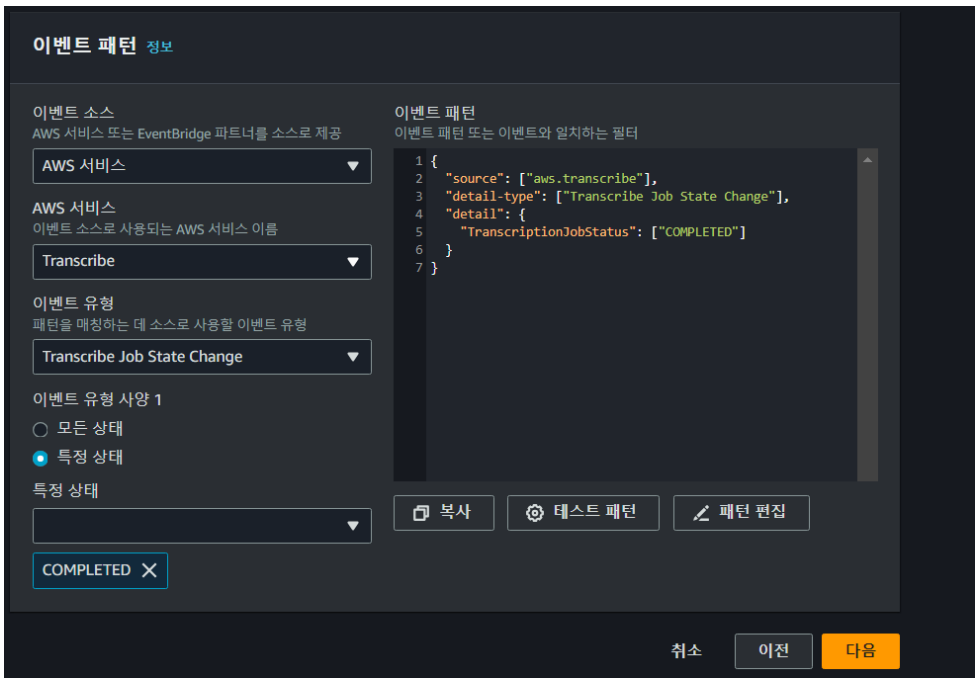


이제 cloudwatch가 AWS Transcribe의 변환 완료 이벤트를 감지해 (eventbridge가) save_text을 호출하는 것을 구축한다.

(1) IAM 생성

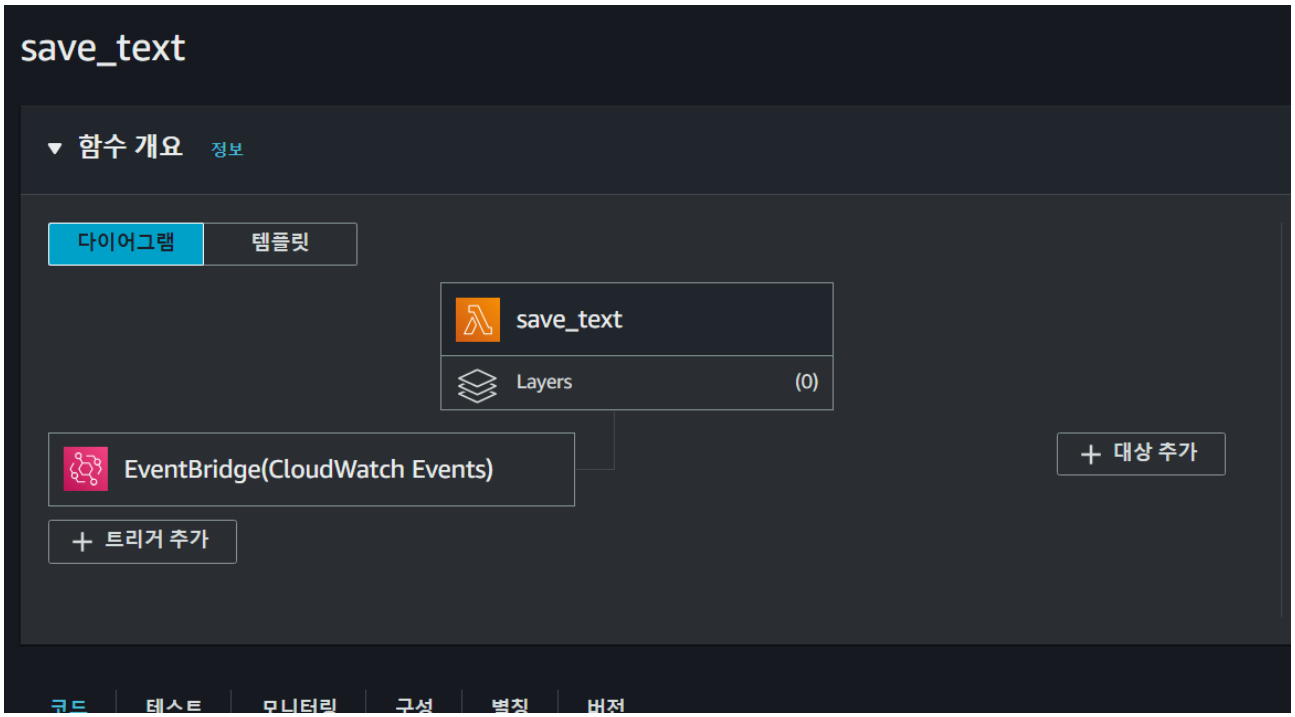


send_email 람다 함수에 적용할 IAM을 생성한다 이 IAM은 RDS, Transcribe, VPC, CloudWatchLog에 접근할 수 있어야 한다.



이제 eventbridge에서 어떤 이벤트를 감지할지 이벤트 패턴을 생성한다. 위와 같이 aws.transcribe 서비스의 작업 상태가 완료(COMPLETED)인 것으로 설정한다.

해당 이벤트 패턴이 발생하면 send_email 람다 함수를 호출하도록 설정한다.



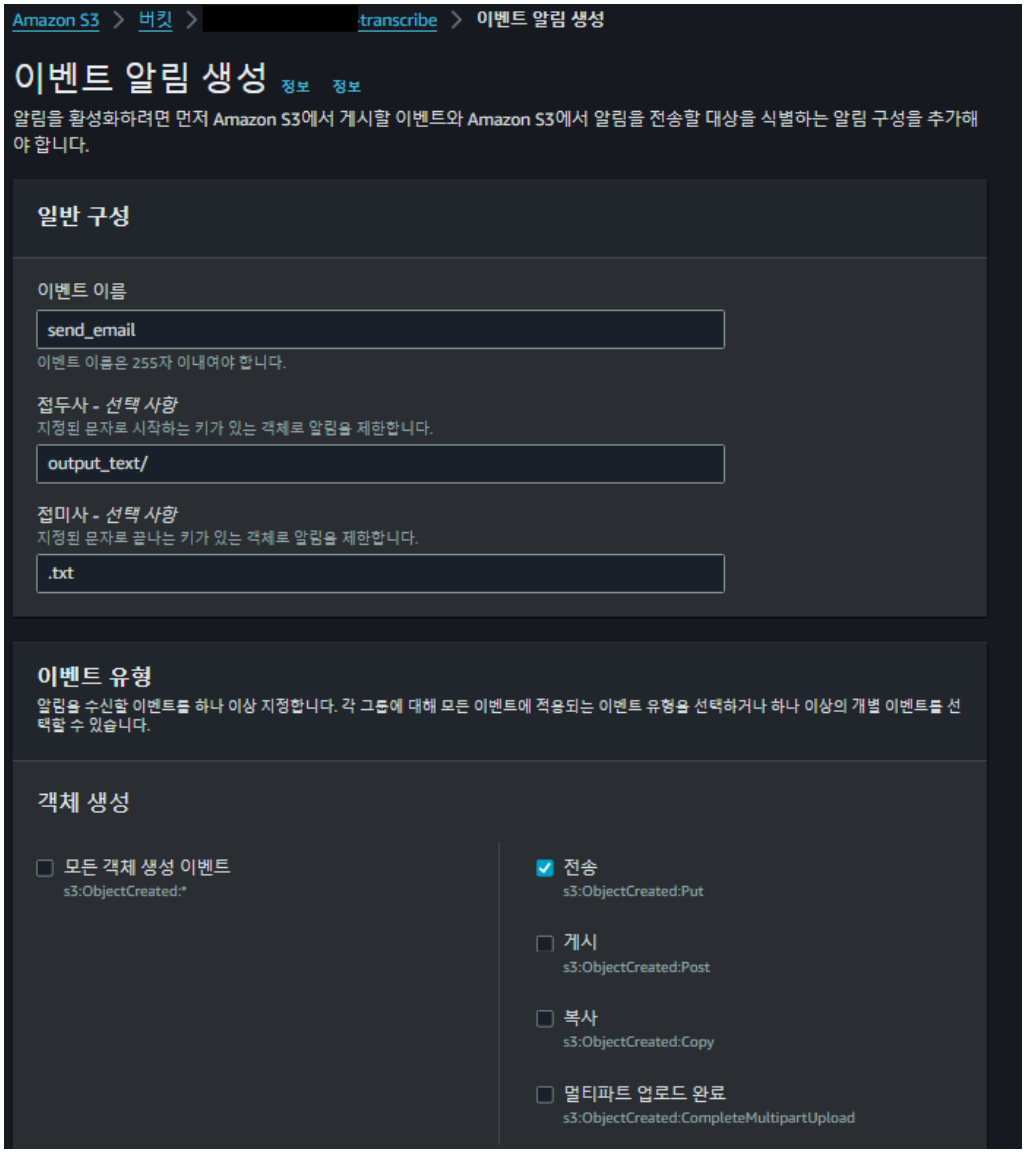
이제 send_email 람다 함수에 EventBridge가 트리거로 설정된 것을 확인할 수 있다.

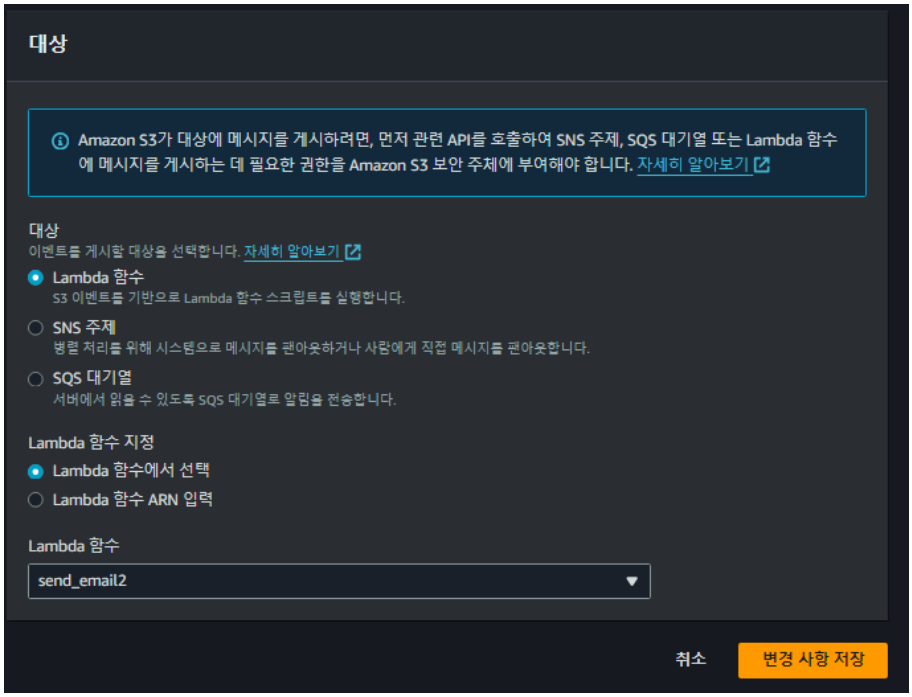
save_text 람다함수의 환경변수를 설정한다.

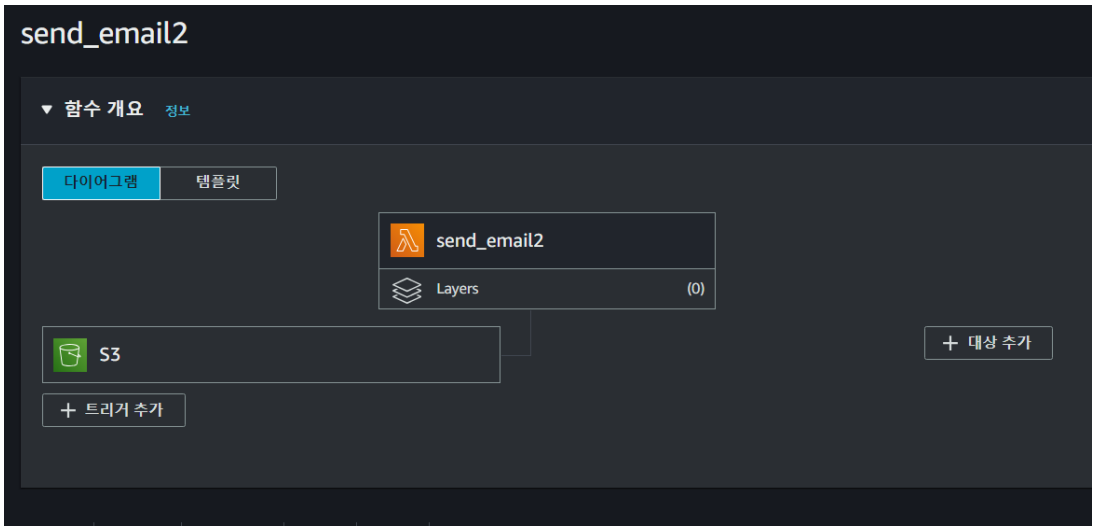
7. send_email 구성

(1) 이벤트 알림 생성

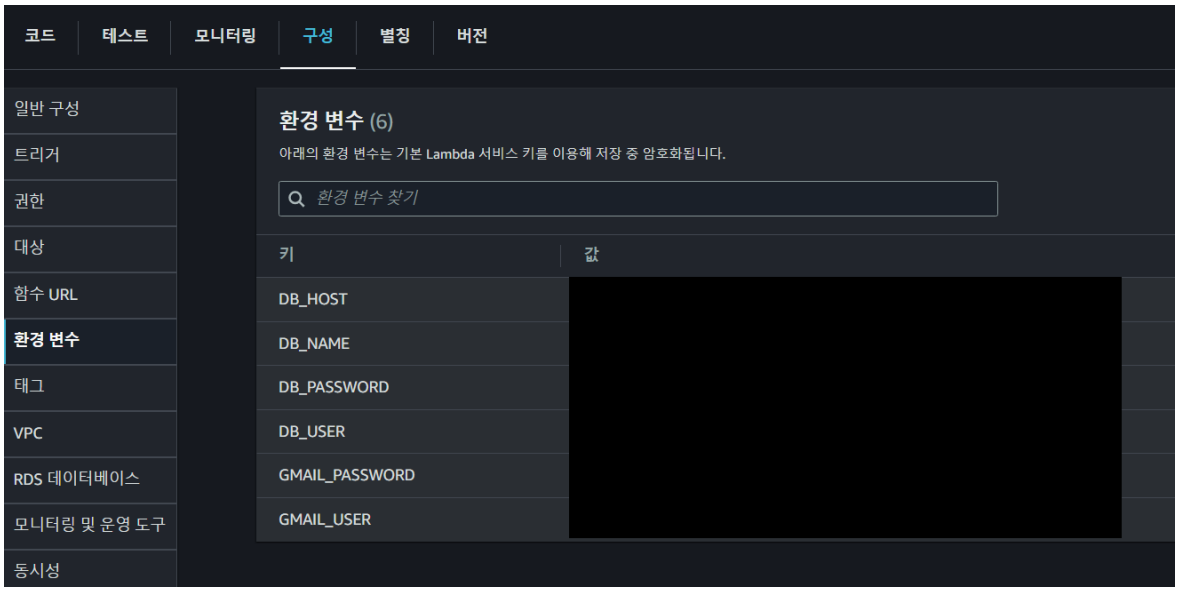
s3 버킷에 텍스트 파일이 업로드되면, 이메일을 보내는 send_email 람다 함수를 트리거하는 것을 구현한다.







위와 같이 S3에 의해 send_email2가 트리거된다.



send_email 람다 함수의 환경변수를 설정한다.

8. 결론

   AWS 콘솔을 이용하여 다양한 AWS 서비스를 생성하고, 서비스들끼리 연결시켰다. AWS를 이용하여 서비스를 구축하면서, AWS에 생각보다 다양한 기능이 있음을 알게 되었다.

9. 참고 자료

   - https://duckgugong.tistory.com/336
   - https://heytech.tistory.com/403
   - https://medium.com/@manishdiddi03/automating-audio-transcription-how-to-use-aws-api-gateway-s3-sns-lambda-and-transcribe-to-20c220b1e77f
   - https://medium.com/analytics-vidhya/transcribing-audio-files-with-amazon-transcribe-lambda-s3-474dc9a1ced7
   - https://velog.io/@chaduri7913/AWS-RDS-Lambda-%EC%97%B0%EA%B2%B0%ED%95%98%EA%B8%B0